

AD-A184 978

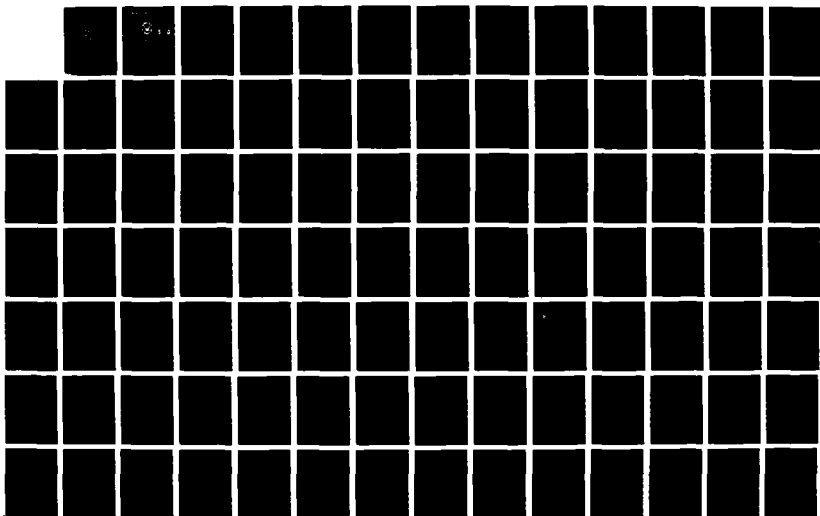
AN ADAPTIVE LATTICE ALGORITHM FOR SPECTRAL LINE
ESTIMATION(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA
I K PARK JUN 87

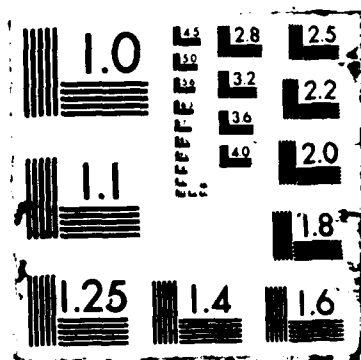
1/2

UNCLASSIFIED

F/G 12/1

ML





AD-A184 970

(2)

NAVAL POSTGRADUATE SCHOOL

Monterey, California

DTIC FILE COPY



DTIC
ELECTE
OCT 07 1987
S D
OK D

THESIS

AN ADAPTIVE LATTICE ALGORITHM FOR
SPECTRAL LINE ESTIMATION

by

Ill Koo Park

June 1987

Thesis Advisor:

Murali Tummala

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY			3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE				
4 PERFORMING ORGANIZATION REPORT NUMBER(S)			5 MONITORING ORGANIZATION REPORT NUMBER(S)	
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School		6b OFFICE SYMBOL (if applicable) 62	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000			7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	
8a NAME OF FUNDING/SPONSORING ORGANIZATION		8b OFFICE SYMBOL (if applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c ADDRESS (City, State, and ZIP Code)			10 SOURCE OF FUNDING NUMBERS	
			PROGRAM ELEMENT NO	PROJECT NO
			TASK NO	WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) AN ADAPTIVE LATTICE ALGORITHM FOR SPECTRAL LINE ESTIMATION				
12 PERSONAL AUTHOR(S) PARK, Ill K.				
13a TYPE OF REPORT Master's Thesis		13b TIME COVERED FROM _____ TO _____	14 DATE OF REPORT (Year, Month, Day) 1987 June	15 PAGE COUNT 103
16 SUPPLEMENTARY NOTATION				
17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Adaptive Lattice Algorithm	
FIELD	GROUP	SUB-GROUP		
19 ABSTRACT (Continue on reverse if necessary and identify by block number) In this thesis we derive a lattice structure to realize linear phase transfer functions and develop an adaptive algorithm for continuously updating the lattice reflection coefficients. The lattice structure is considered because of its superior finite wordlength performance compared to transversal structures. The adaptive lattice algorithm developed in this thesis has been applied to estimate the sinusoidal frequencies as part of Prony's method. Results of computer simulation supporting the theory are reported.				
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS			21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL Murali Tummala			22b TELEPHONE (Include Area Code) 408-646-2645	22c OFFICE SYMBOL 62Tu

Approved for public release; distribution is unlimited.

An Adaptive Lattice Algorithm for Spectral Line Estimation

by

Ill Koo Park
Lieutenant Commander, Republic of Korea Navy
B.S., R.O.K. Naval Academy, 1978
B.S., R.O.K. Yonsei University, 1983

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

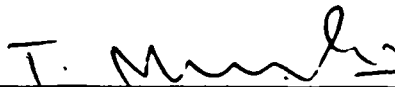
NAVAL POSTGRADUATE SCHOOL
June 1987

Author:

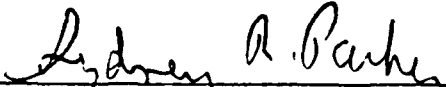


Ill Koo Park

Approved by:



Murali Tummala, Thesis Advisor



Sydney R. Parker, Second Reader



John P. Powers, Chairman,
Department of Electrical and Computer Engineering



Gordon E. Schacher,
Dean of Science and Engineering

ABSTRACT

In this thesis we derive a lattice structure to realize linear phase transfer functions and develop an adaptive algorithm for continuously updating the lattice reflection coefficients. The lattice structure is considered because of its superior finite wordlength performance compared to transversal structures. The adaptive lattice algorithm developed in this thesis has been applied to estimate the sinusoidal frequencies as part of Prony's method. Results of computer simulation supporting the theory are reported.

TABLE OF CONTENTS

I.	INTRODUCTION	10
II.	PRONY'S SPECTRAL LINE ESTIMATION	12
	A. INTRODUCTION	12
	B. PRONY'S METHOD	12
	C. ESTIMATION OF FREQUENCY, AMPLITUDE AND PHASE	15
III.	LINEAR PHASE FIR FILTERING	19
	A. INTRODUCTION	19
	B. LINEAR PHASE FIR FILTERS [REF. 3]	19
	C. LATTICE FILTERS [REF. 4]	22
	D. LINEAR PHASE FIR LATTICE FILTER	27
	E. LATTICE REALIZATION OF A GENERAL FIR TRANSFER FUNCTION	31
IV.	LMS ALGORITHM	34
	A. INTRODUCTION	34
	B. SUMMARY OF THE LMS ALGORITHM	35
	C. ADAPTIVE LATTICE ALGORITHMS	39
	1. Non-adaptive Methods	41
	2. Adaptive Methods	42
	D. SIMULATION RESULTS	45
V.	ADAPTIVE LINEAR PHASE LATTICE ALGORITHM	48
	A. INTRODUCTION	48
	B. LMS ALGORITHM FOR THE FIR LATTICE	48
	C. LINEAR PHASE FIR LATTICE ALGORITHM	58
	D. SPECTRAL ESTIMATION	64
	E. SUMMARY, CONCLUSION AND SUGGESTED FUTURE WORK	74

APPENDIX A: EXAMPLES OF OBTAINING A LATTICE STRUCTURE FOR THE FIR TRANSFER FUNCTION	75
APPENDIX B: COMPUTER PROGRAMS	81
LIST OF REFERENCES	100
INITIAL DISTRIBUTION LIST	102



Accession For	
NTIS	<input checked="" type="checkbox"/>
DTIC	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution /	
Availability Codes	
Dist	Availability Codes
A-1	

LIST OF TABLES

1. LMS ALGORITHM FOR THE FIR LATTICE 56
2. LINEAR PHASE FIR LATTICE ALGORITHM 61

LIST OF FIGURES

3.1	Direct-form realization for an FIR filter with linear phase ($N = \text{even}$)	21
3.2	Direct-form realization for an FIR filter with linear phase ($N = \text{odd}$)	22
3.3	Lattice Section	24
3.4	Linear Phase FIR Lattice Filter	28
3.5	FIR Lattice	32
4.1	Parallel Form	35
4.2	Serial Form	36
4.3	A Single Stage of Lattice Filter	40
4.4	Lattice Form Implementation of Noise-cancelling Filter	44
4.5	System Identification Experiment	45
4.6	Learning Curve ($\mu = 0.01$)	46
4.7	Learning Curve ($\mu = 0.1$)	47
4.8	Learning Curve ($\mu = 0.5$)	47
5.1	Adaptive FIR Lattice Filter	49
5.2	Computation of Gradient Elements for the Adaptive FIR Lattice Algorithm	53
5.3	Learning Curve ($\mu = 0.1$)	57
5.4	Learning Curve ($\mu = 0.3$)	57
5.5	Learning Curve ($\mu = 0.5$)	58
5.6	Adaptive Linear Phase FIR Lattice Filter	59
5.7	Learning Curve ($\mu = 0.1$)	62
5.8	Learning Curve ($\mu = 0.3$)	62
5.9	Learning Curve ($\mu = 0.03$)	63
5.10	Learning Curve ($\mu = 0.1$)	63
5.11	Spectral Estimation Mode	66
5.12	Frequency Spectrum (1 sinusoid, $M = 2$, $\mu = 0.015$)	68
5.13	Frequency Spectrum (1 sinusoid, $M = 4$, $\mu = 0.01$)	69
5.14	Frequency Spectrum (1 sinusoid, $M = 10$, $\mu = 0.03$)	69

5.15	Frequency Spectrum (1 sinusoid, $M = 20$, $\mu = 0.03$)	70
5.16	Frequency Spectrum (1 sinusoid, $M = 20$, SNR = 30dB)	70
5.17	Frequency Spectrum (1 sinusoid, $M = 20$, SNR = 20dB)	71
5.18	Frequency Spectrum (1 sinusoid, $M = 20$, SNR = 10dB)	71
5.19	Frequency Spectrum (2 sinusoids, $M = 4$, $\mu = 0.02$)	72
5.20	Frequency Spectrum (2 sinusoids, $M = 20$, $\mu = 0.022$)	72
5.21	Frequency Spectrum (4 sinusoids, $M = 8$, $\mu = 0.015$)	73
5.22	Frequency Spectrum (4 sinusoids, $M = 30$, $\mu = 0.014$)	73

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation for the support, guidance and continuous encouragement in completing this thesis given by my thesis advisor, Professor Murali Tummala, and my second reader Professor Sydney R. Parker, of the Electrical and Computer Engineering Department of the Naval Postgraduate School.

Finally, I offer a very special thanks to my wife, Kyung Sook, my son, Hae Won, and my daughter, Cho Won, for their patience and for their support away from our home, during these two years in the United States.

I. INTRODUCTION

Every finite-impulse response (FIR) filter has two distinct properties [Ref. 5] ; first, it is always stable; second, if it is not causal, it can always be made to be causal by introducing a finite delay. FIR filters can be designed so that their frequency responses have an exactly linear phase characteristic. FIR filters with linear phase are important in applications like speech processing and data transmission, because in these applications a nonlinear phase filter is harmful. The symmetry property of the linear phase FIR filters, however, helps reduce the number of coefficients by nearly one half resulting in substantial computational savings.

The various filter realizations, or structures that are frequently considered are the direct form, the cascade form, the parallel form, and the lattice form. The lattice form realization is of particular interest because of its superior numerical performance, and modularity in the structure. The operation of a lattice filter is completely described by specifying the sequence of reflection coefficients that characterize the individual stages of the filter.

Conventionally an adaptive filter is composed of a tapped delay line or transversal structure with adjustable coefficients or weights and an adaptive algorithm which updates the coefficients continuously based on some performance criterion. The design of a fixed coefficient filter is based on the prior knowledge of both signal and noise. Adaptive filters, on the other hand, have the ability to adjust their own parameters automatically, and their design requires little, or no *a priori* knowledge of signal or noise characteristics [Ref. 14]. However, the designer has to choose the order of the filter and the type of the algorithm. Also the adaptive filter usually requires a large initial transient time (i.e., the initial filter convergence period).

The least-mean-square (LMS) adaptive algorithm minimizes the mean square error $e(k)$ by recursively altering the filter coefficient vector $\underline{z}(k)$ at each sampling instant. The original Widrow-Hoff LMS algorithm is $\underline{z}(k+1) = \underline{z}(k) - 2\mu e(k)\underline{z}(k)$, where μ is a convergence factor controlling stability and rate of adaptation [Ref. 15]. The algorithm is based on the method of steepest descent, moving $b(k)$ in proportion to the instantaneous gradient estimate of the mean square error. The successive orthogonalization provided by the lattice offers advantages in adaptive convergence rate which cannot be achieved with tapped delay lines [Ref. 17,18].

Recently, Prony's method has been applied to the estimation of spectral lines in noise [Ref. 11]. It has been shown that the Prony's method yields better spectral estimates than a companion spectral estimation technique, called Pisarenko's method [Ref. 10]. Also it has been established that the filter structure involved in Prony's method has a linear phase property which is not the case for Pisarenko's method [Ref. 11].

The thesis investigates the application of lattice structures in Prony's method of spectral line estimation. The complete solution to Prony's method consists of three steps: (i) representing a given process of M sinusoids in noise in terms of complex exponentials, (ii) finding roots of a symmetric polynomial, and (iii) estimating the frequency, phase and amplitude information. In this thesis, however, we have emphasized only the frequency estimation problem. Here we derive an adaptive lattice structure to realize linear phase transfer functions which will be used to estimate the sinusoidal frequencies as part of Prony's method. The scope of the thesis consists of obtaining a linear phase lattice structure, developing a least mean square (LMS) error based adaptive algorithm, and testing the lattice structure and the algorithm by means of computer simulation.

The thesis is organized as follows. In Chapter II, we present a brief review of Prony's method of representing a given process in terms of a set of complex exponentials, and then address the problem of estimating spectral lines using this method. We show that the original Prony's method has to be modified slightly in order to apply it to the spectral line estimation problems. In Chapter III, we discuss the basic concepts of the linear phase FIR filter and the related lattice structure. We show three examples of obtaining the lattice structures for both linear phase and non-linear phase FIR transfer functions (Appendix A). In Chapter IV, we briefly discuss the least-mean-square (LMS) adaptive algorithm that results from attempting to minimize the mean square error and present a summary of some LMS algorithms for lattice that have been reported in the literature. Also included are some computer simulation results. In Chapter V, we present the derivation of a new LMS based FIR adaptive lattice algorithm and extend it to the linear phase case as well. The new algorithm is then used in the estimation of spectral lines in white noise. Results of simulation are included.

II. PRONY'S SPECTRAL LINE ESTIMATION

A. INTRODUCTION

In its original form, Prony's method analyzes processes involving simple, or damped harmonics using complex exponential functions as coordinate functions. On the other hand, the well known Fourier analysis consists of representing a given process in terms of a set of sine and cosine functions. Recently, Prony's method has been applied to the estimation of spectral lines in noise [Ref. 11]. It has been shown that the Prony's method yields better spectral estimates than a companion spectral estimation technique, called Pisarenko's method, in terms of bias, spurious responses, and the computational complexity [Ref. 10].

In this chapter, we present a brief review of the Prony's method of representing a given process in terms of a set of complex exponentials, and then address the problem of estimating spectral lines using this method. We show that original Prony's method has to be modified slightly in order to apply it to the spectral line estimation problems.

B. PRONY'S METHOD

Consider that the given process, $x(k)$, consists of n distinct sinusoids, then $x(k)$ can be approximated by an expression of the form

$$x(k) = \sum_{i=1}^n [A_i \cos \omega_i k + B_i \sin \omega_i k] \quad (2.1)$$

where the ω_i 's are sinusoidal frequencies. The above approximation can be considered as a special case of an exponential approximation given by [Ref. 29]

$$x(k) = \sum_{i=1}^m C_i e^{ja_i k} \quad (2.2)$$

where $m = 2n$ and the a_i 's are identified as ω_i and $-\omega_i$. The values of ω_i can be estimated, by Prony's method, assuming that the data are known at $k = 0, 1, \dots, N-1$. From Eq.(2.2), we can obtain the following set of equalities

$$C_1 + C_2 + \dots + C_m = x(0)$$

$$C_1 e^{ja_1} + C_2 e^{ja_2} + \dots + C_m e^{ja_m} = x(1)$$

$$C_1 e^{j2a_1} + C_2 e^{j2a_2} + \dots + C_m e^{j2a_m} = x(2) \quad (2.3)$$

⋮

$$C_1 e^{j(N-1)a_1} + C_2 e^{j(N-1)a_2} + \dots + C_m e^{j(N-1)a_m} = x(N-1)$$

Now we have a set of N equations with $2m$ unknowns, namely, C_i and a_i ($i = 1, 2, \dots, m$) which can be solved exactly if $N = 2m$, or approximately by the method of least squares if $N > 2m$. Also note that the N equations are nonlinear in the exponential terms e^{ja_i} . Let e^{ja_i} , $i = 1, 2, \dots, m$, be the roots of the equation [Ref. 29].

$$e^{jma} + \alpha_1 e^{j(m-1)a} + \alpha_2 e^{j(m-2)a} + \dots + \alpha_{m-1} e^{ja} + \alpha_m = 0 \quad (2.4)$$

In order to determine the coefficients α_i ($i = 1, 2, \dots, m$), we multiply the first equation in Eq.(2.3) by α_m , the second equation by α_{m-1} , ..., the m^{th} equation by α_1 and the $(m+1)^{\text{th}}$ equation by 1, and add the results. In this way, we can obtain the $(N-m)$ linear equations. From Eq.(2.2), we can obtain the following set of equalities

$$x(m) + x(m-1) \alpha_1 + x(m-2) \alpha_2 + \dots + x(0) \alpha_m = 0$$

$$x(m+1) + x(m) \alpha_1 + x(m-1) \alpha_2 + \dots + x(1) \alpha_m = 0$$

(2.5)

⋮

$$x(N-1) + x(N-2) \alpha_1 + x(N-3) \alpha_2 + \dots + x(N-m-1) \alpha_m = 0$$

Since the data $x(k)$, $k = 0, 1, 2, \dots, N-1$, are known, this set of equations can be solved for the m α 's if $N \geq 2m$. After the α 's are determined, e^{ja_i} , $i = 1, 2, \dots, m$, are found from Eq.(2.4). The set of equations in Eq.(2.3) then becomes linear in terms of C_i , $i = 1, 2, \dots, m$. The C 's can be determined from the first m equations, or determined approximately by applying the least squares method to the entire set of equations. The nonlinearity associated in finding e^{ja_i} , which is related to the frequencies $j\omega_i$ and $-j\omega_i$, is concentrated in Eq.(2.4). The above procedure is known as Prony's method.

Now, in the case of Eq.(2.1), since the roots of Eq.(2.4) occur in reciprocal pairs, Eq.(2.4) must be invariant under the substitution of e^{-ja_i} for e^{ja_i} , so that we must have $\alpha_{2n} = 1$, $\alpha_{(2n-1)} = \alpha_1, \dots, \alpha_{n+1} = \alpha_{n-1}$. Thus Eq.(2.4) becomes

$$e^{j2n\omega} + \alpha_1 e^{j(2n-1)\omega} + \dots + \alpha_{n-1} e^{j(n+1)\omega} + \alpha_n e^{jn\omega} \\ + \alpha_{n-1} e^{j(n-1)\omega} + \dots + \alpha_1 e^{j\omega} + 1 = 0$$

or

$$e^{jn\omega} [(e^{jn\omega} + e^{-jn\omega}) + \alpha_1 (e^{j(n-1)\omega} + e^{-j(n-1)\omega}) + \dots + \\ \alpha_{n-1} (e^{j\omega} + e^{-j\omega}) + \alpha_n] = 0$$

since $e^{jn\omega} \neq 0$, we have

$$2 \cos n\omega + 2 \alpha_1 \cos(n-1)\omega + \dots + 2 \alpha_{n-1} \cos \omega + \alpha_n = 0 \quad (2.6)$$

Now noting that $m=2n$, and applying the above symmetry of α 's to Eq.(2.5) yields [Ref. 11]

$$\{x(0) + x(2n)\} + \{x(1) + x(2n-1)\} \alpha_1 + \dots + \{x(n-1) + x(n+1)\} \alpha_{n-1}$$

$$+ x(n) \alpha_n = 0$$

$$\{x(1) + x(2n+1)\} + \{x(2) + x(2n)\} \alpha_1 + \dots + \{x(n) + x(n+2)\} \alpha_{n-1}$$

$$+ x(n+1) \alpha_n = 0$$

(2.7)

$$\{x(N-2n-1) + x(N-1)\} + \{x(N-2n) + x(N-2)\} \alpha_1 + \dots + \{x(N-n-2)$$

$$+ x(N-n)\} \alpha_{n-1} + x(N-n-1) \alpha_n = 0$$

Eq.(2.7) consists of a set of $N-2n$ equations and n unknowns. This set can be solved exactly for the α 's if $N=3n$, or solved approximately by the least squares method if $N>3n$, and then the ω 's are determined from Eq.(2.6).

C. ESTIMATION OF FREQUENCY, AMPLITUDE AND PHASE

If a process under measurement contains an unknown number of sinusoids of unknown frequencies and amplitudes, a variant of Prony's method can be used to determine the number of sinusoids and their associated frequencies and amplitudes. As noted above, Prony's method is a technique for modeling data of equally spaced samples by a linear combination of exponentials. An exponential curve having M exponentials terms can be determined from the $2M$ data measurements. Each exponential term $A_i e^{a_i k}$ has two parameters — an amplitude A_i and an exponent a_i , where a_i can be real or imaginary. For the case where only an approximate fit with M exponentials to a data set of N samples is desired, such that $N>2M$, a least squares estimation procedure is used.

The model assumed is a set of M exponentials of arbitrary amplitude, phase, frequency and damping factor. A process consisting of M real undamped (a is imaginary) sinusoids can be expressed as

$$\begin{aligned} x(k) &= \sum_{i=1}^M (a_i z_i^k + a_i^* z_i^{*k}) \\ &= \sum_{i=1}^M A_i \cos(2\pi f_i kT + \theta_i) \end{aligned} \quad (2.8)$$

with $a_i = (A_i/2)e^{j\theta_i}$ and $z_i = e^{j2\pi f_i T}$, where A_i is the amplitude, f_i is the frequency and θ_i is the phase of the i^{th} sinusoid, respectively, and T is the sampling interval.

Finding $\{A_i, \theta_i, f_i\}$ and M that minimize the squared error is a difficult nonlinear least squares problem. Prony's method solves two sequential sets of linear equations with an intermediate polynomial rooting step that concentrates the nonlinearity of the problem in the polynomial rooting procedure (similar to Eq.(2.4)).

Define the polynomial, $B(z)$, which has z_i and z_i^* as its roots, given by

$$B(z) = \prod_{i=1}^M (z - z_i)(z - z_i^*) = \sum_{j=0}^{2M} b_j z^{2M-j} = 0 \quad (2.9)$$

with $b_0 = 1$ and the b_j being real coefficients. Since the roots are of unit modulus occur in complex conjugate pairs, then Eq.(2.9) must be invariant under the substitution z^{-1} for z . Therefore, Eq.(2.9) can be written as

$$z^{2M} B(1/z) = z^{2M} \sum_{j=0}^{2M} b_j z^{j-2M} = \sum_{j=0}^{2M} b_j z^j = 0 \quad (2.10)$$

Comparing Eqs.(2.9) and (2.10), we may conclude that $b_j = b_{2M-j}$ for $j=0,1, \dots, M$, with $b_0 = b_{2M} = 1$. Thus the requirement for complex conjugate root pairs of unit modulus is implemented by constraining the polynomial coefficients to be symmetric about the center element. Hence the realization of $B(z)$ is a linear phase FIR filter. Based on order $2M$, a linear prediction error can be written as

$$\varepsilon(k) = \sum_{j=0}^M b_j [x(k+j) + x(2M-k+j)] \quad (2.11)$$

which reduces the number of coefficients required by one-half.

The coefficients b_1, b_2, \dots, b_m are determined in a least squares fashion by minimizing the total squared error.

$$E = \sum_{k=0}^{N-2M} \varepsilon^2(k) \quad (2.12)$$

which yields the normal equations

$$\sum_{j=0}^m b_k \left[\sum_{i=0}^{N-2M} [x(2M-k+i) + x(k+i)] [x(2M-j+i) + x(j+i)] \right] = 0 \quad (2.13)$$

for $k=1, \dots, M$

This equation can be solved recursively for increasing order M .

From the estimated $\{b_j\}$ values, the $\{z_i\}$ are determined using Eq.(2.9). This gives the frequency estimates.

$$f_i = \tan^{-1} [\text{Im}(z_i)/\text{Re}(z_i)] / 2\pi T \quad (2.14)$$

To obtain the $\{\alpha_i\}$ a second set of normal equations is solved,

$$\begin{aligned} & \sum_{i=1}^M \left[\alpha_i \sum_{j=0}^N z_k^j z_i^j + \alpha_i^* \sum_{j=0}^N z_k^{*j} z_i^{*j} \right] \\ & = \sum_{j=0}^N (2 \text{Re } z_k^j) x(j) \end{aligned} \quad (2.15)$$

for $k=0,1, \dots, M$

The set $\{\alpha_i\}$ provides both amplitude (A_i), or power, and phase (θ_i) information:

$$A_i = |\alpha_i| \quad (2.16)$$

$$\theta_i = \tan^{-1} [\text{Im}(b_i) / \text{Re}(b_i)] \quad (2.17)$$

In the foregoing we have shown that the frequency estimation as part of Prony's method requires a polynomial with a linear phase property. In the next chapter, we show that a lattice structure can be utilized to implement a linear phase transfer function.

III. LINEAR PHASE FIR FILTERING

A. INTRODUCTION

Every finite-impulse response (FIR) filter has two distinct properties [Ref. 5] ; first, it is always stable; second, if it is not causal, it can always be made to be causal by introducing a finite delay.

FIR filters can be designed so that their frequency responses have an exactly linear phase characteristic. FIR filters with linear phase are important in applications like speech processing and data transmission, because in these applications a nonlinear phase filter is harmful.

The impulse response sequence of a linear phase FIR filter exhibits a kind of symmetry, e.g., $h(n) = h(N-1-n)$ for $n = 0, 1, \dots, (N/2)-1$ (assuming that N is even). In general, FIR filters require a large number of coefficients to adequately meet with the required filter specifications. The symmetry property of the linear phase FIR filters, however, helps reduce the number of coefficients by nearly one half resulting in substantial computational savings.

The various filter realizations, or structures that are frequently considered are the direct form, the cascade form, the parallel form, and the lattice form. The lattice form realization is of particular interest because of its superior numerical performance, and modularity in the structure. Lattice realizations have been successfully utilized in filtering and spectral analysis, and in modeling of some physical process like speech, geophysical data etc. [Ref. 8]. The operation of a lattice filter is completely described by specifying the sequence of reflection coefficients that characterize the individual stages of the filter.

In this chapter, we will discuss the basic concepts of the linear phase FIR filter and the related lattice structure. And finally, we will show three examples of obtaining the lattice structures realizing for both phase and non-linear phase FIR transfer functions (Appendix A).

B. LINEAR PHASE FIR FILTERS [REF. 3]

Let $\{h(n)\}$ be a causal finite duration sequence defined over the interval $0 \leq n \leq N-1$ having the linear phase symmetry property given by $h(n) = h(N-1-n)$. The z transform of $\{h(n)\}$ can be written as

$$H(z) = \sum_{n=0}^{N-1} h(n)z^{-n} \quad (3.1)$$

If N is even, then Eq.(3.1) can be rewritten as

$$H(z) = \sum_{n=0}^{(N/2)-1} h(n)z^{-n} + \sum_{n=N/2}^{N-1} h(n)z^{-n}$$

Now applying the symmetry property of $h(n)$ in the second term on the right hand side yields

$$H(z) = \sum_{n=0}^{(N/2)-1} h(n)z^{-n} + \sum_{n=0}^{(N/2)-1} h(N-1-n)z^{-(N-1-n)}$$

which can be simplified to

$$H(z) = \sum_{n=0}^{(N/2)-1} h(n) \{z^{-n} + z^{-(N-1-n)}\} \quad (3.2)$$

If N is odd, then Eq.(3.1) becomes

$$H(z) = \sum_{n=0}^{[(N-1)/2]-1} h(n)[z^{-n} + z^{-(N-1-n)}] + h\left(\frac{N-1}{2}\right) z^{-[(N-1)/2]} \quad (3.3)$$

If we evaluate Eqs.(3.2) and (3.3) for $z = e^{j\omega}$, we obtain the discrete Fourier transform of the filter sequence $h(n)$ defined as

$$H(e^{j\omega}) = \sum_{n=0}^{N-1} h(n) e^{-j\omega n} \quad (3.4)$$

For the case when N is even, we then have the discrete Fourier transform

$$H(e^{j\omega}) = e^{-j\omega\{(N-1)/2\}} \left[\sum_{n=0}^{(N/2)-1} 2h(n) \cos[\omega\{n-(N-1)/2\}] \right] \quad (3.5)$$

and for N odd, we obtain

$$H(e^{j\omega}) = e^{-j\omega\{(N-1)/2\}} \left[h\{(N-1)/2\} + \sum_{n=0}^{(N-3)/2} 2h(n) \cos[\omega\{n-(N-1)/2\}] \right]. \quad (3.6)$$

In both cases above, the sums in brackets are real, implying a linear phase shift corresponding to a delay of $(N-1)/2$ samples. Figures 3.1 and 3.2 show the direct form realizations of an FIR filter with linear phase for both N even and N odd cases.

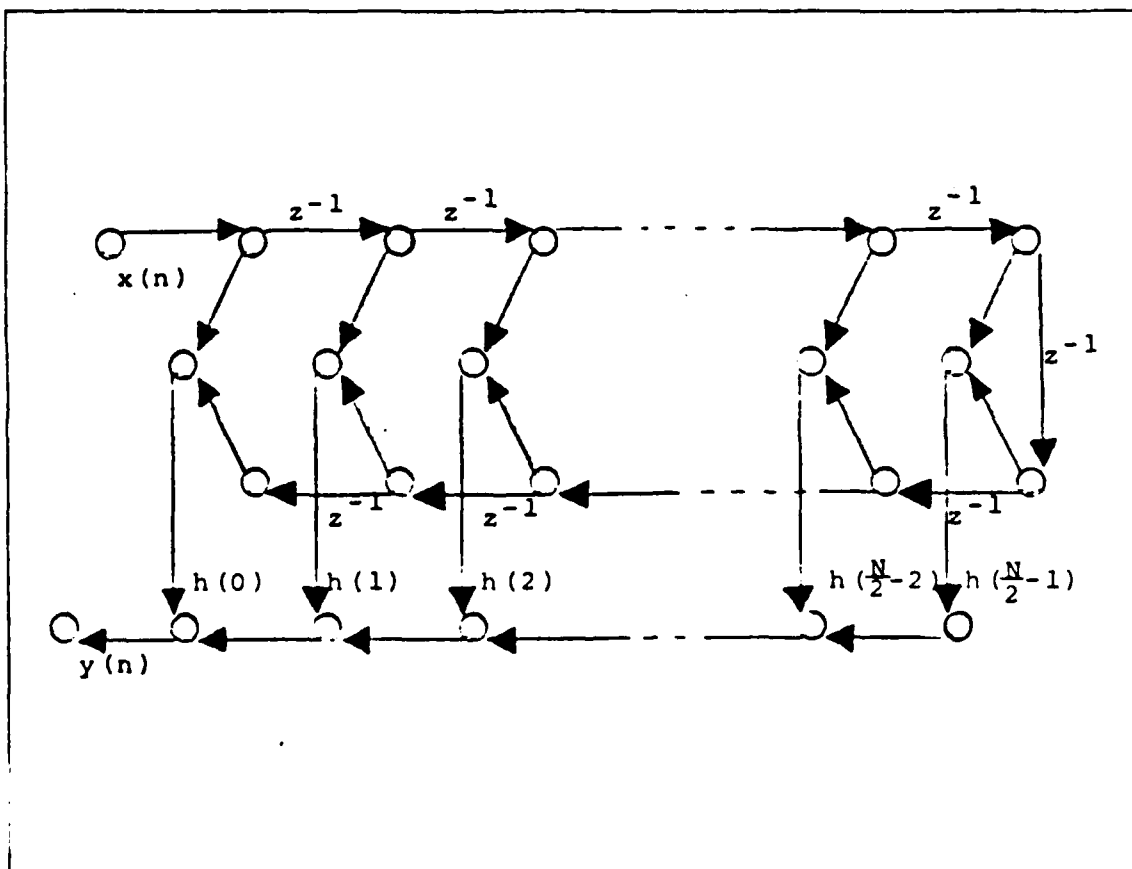


Figure 3.1 Direct-form realization for an FIR filter with linear phase ($N = \text{even}$).

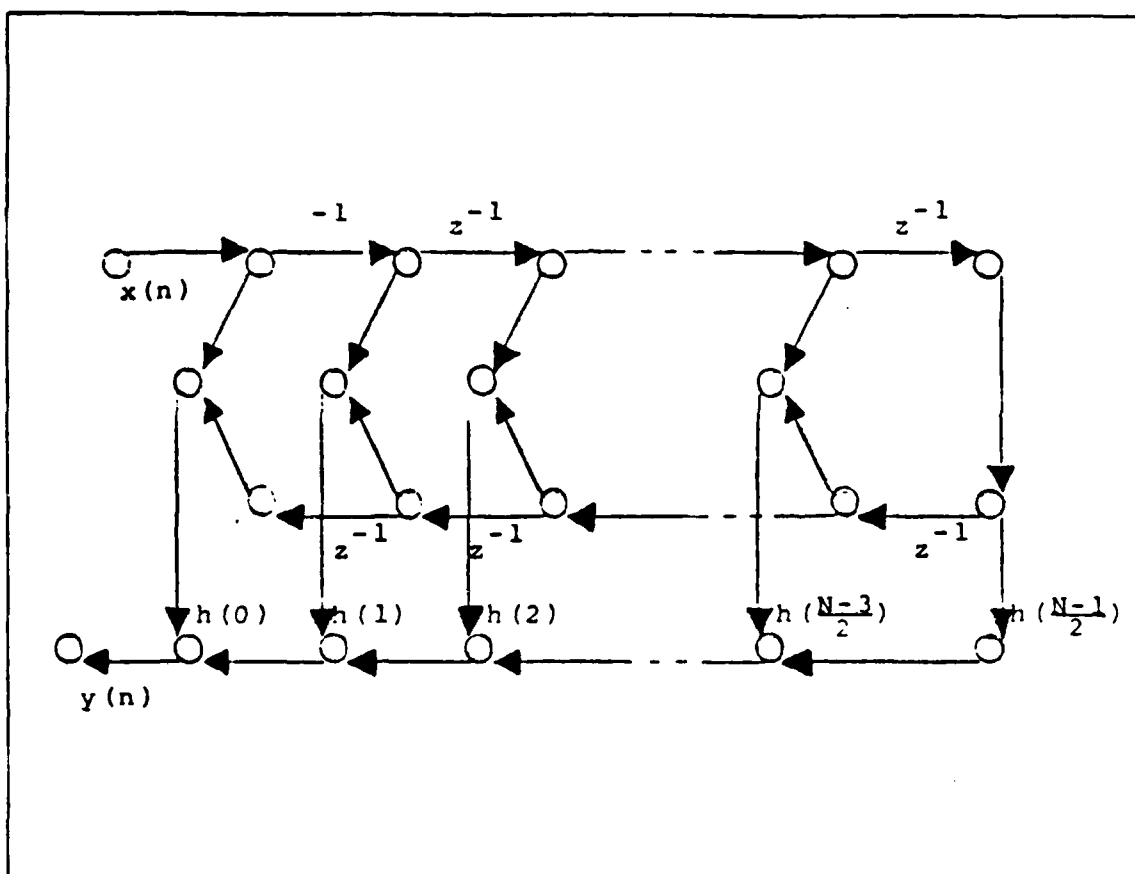


Figure 3.2 Direct-form realization for an FIR filter with linear phase ($N = \text{odd}$).

C. LATTICE FILTERS [REF. 4]

The basic single section lattice structure is shown in Figure 3.3 where $x(n)$ is the input, $f_1(n)$ and $g_1(n)$ are generally known as the forward and backward prediction errors, respectively. The defining equations of the lattice are given by

$$\begin{aligned} f_0(n) &= g_0(n) = x(n) \\ f_1(n) &= f_0(n) + k_1 g_0(n-1) \\ g_1(n) &= k_1 f_0(n) + f_0(n-1) \end{aligned} \quad (3.7)$$

where k_1 is called the lattice reflection coefficient. If another section is cascaded with the first one the resulting equations for the next order prediction errors are given by

$$f_2(n) = f_1(n) + k_2 g_1(n-1)$$

$$g_2(n) = k_2 f_1(n) + g_1(n-1)$$

Substituting for $f_1(n)$ and $g_1(n-1)$ from Eq.(3.7) yields

$$f_2(n) = x(n) + \{k_1 + k_1 k_2\} x(n-1) + k_2 x(n-2) \quad (3.8)$$

or

$$f_2(n) = x(n) + b_{2,1} x(n-1) + b_{2,2} x(n-2)$$

and

$$g_2(n) = k_2 x(n) + \{k_1 + k_1 k_2\} x(n-1) + x(n-2) \quad (3.9)$$

or

$$g_2(n) = b_{2,2} x(n) + b_{2,1} x(n-1) + x(n-2)$$

where $b_{2,1} = k_1(1 + k_2)$ and $b_{2,2} = k_2$.

Thus, by induction, for a cascade connection of M lattice sections we have the general expressions

$$\begin{aligned} f_M(n) &= \sum_{i=0}^M b_{M,i} x(n-i) \\ g_M(n) &= \sum_{i=0}^M b_{M,M-i} x(n-i) \end{aligned} \quad (3.10)$$

where $b_{M,0} = 1$. Eq.(3.10) represents the FIR filter type equations to obtain the M^{th} order forward and backward prediction errors. Now taking the z-transform of Eq.(3.10) yields

$$F_M(z) = \sum_{i=0}^M b_{M,i} z^{-i} X(z) \quad (3.11)$$

$$G_M(z) = \sum_{i=0}^M b_{M,M-i} z^{-i} X(z) \quad (3.12)$$

For an unit impulse input, i.e., when $X(z) = 1$, $F_M(z)$ and $G_M(z)$ represent the forward and backward prediction error filter transfer functions, respectively, and

$$G_M(z) = z^{-M} F_M(z^{-1}) \quad (3.13)$$

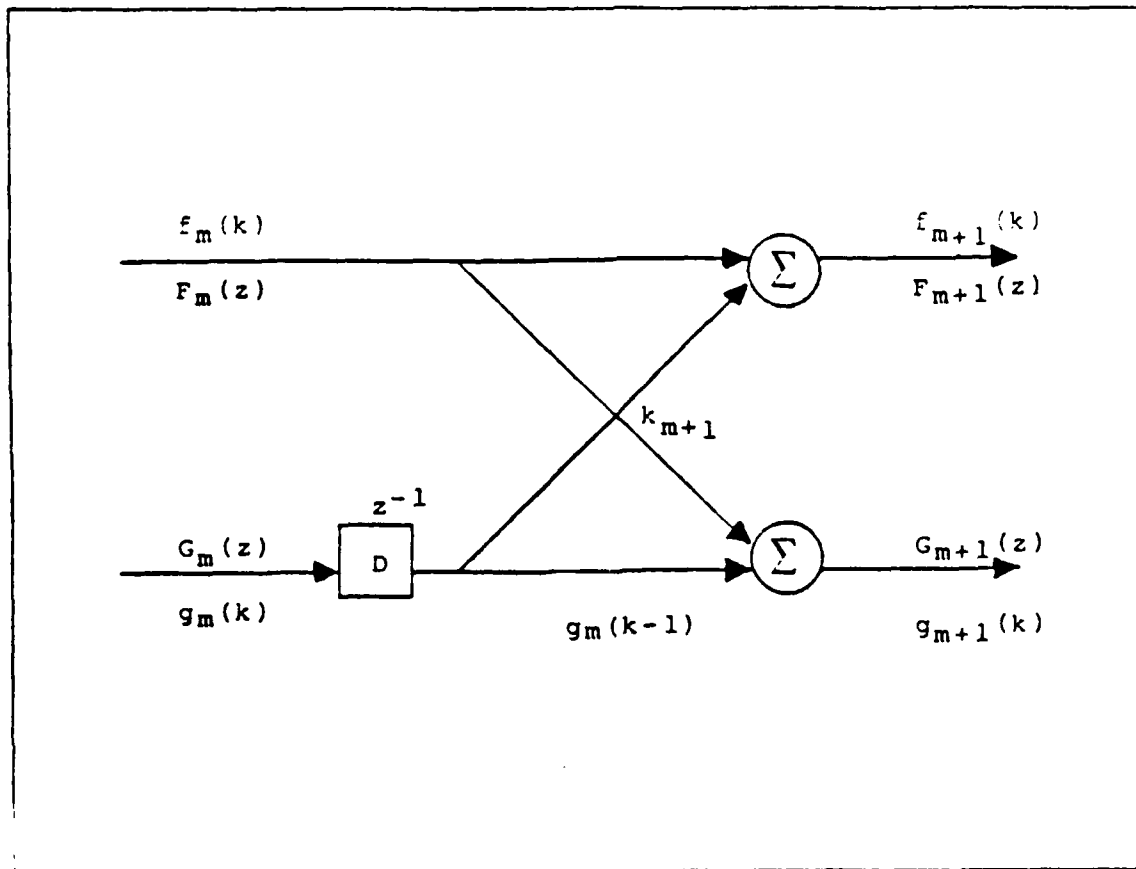


Figure 3.3 Lattice Section.

By inspection of Figure 3.3 we see that the M^{th} stage lattice reflection coefficient is equal to the FIR filter coefficient $b_{M,M}$ in Eqs.(3.11) and (3.12). In other words, we have

$$k_M = b_{M,M} \quad (3.14)$$

In fact, the FIR like prediction error filter coefficients can be iteratively calculated starting from Eq.(3.14). The algorithm, to calculate the filter coefficients $b_{M,i}$, also makes use of the well known lattice property that an M^{th} order lattice contains all prediction filters of orders $m \leq M$. Now consider the m^{th} section ($1 \leq m \leq M$) in the cascade connection of M lattice sections which can be described by the following equations:

$$F_m(z) = F_{m-1}(z) + k_m z^{-1} G_{m-1}(z) \quad (3.15)$$

$$G_m(z) = k_m F_{m-1}(z) + z^{-1} G_{m-1}(z) \quad (3.16)$$

Eq.(3.16) can be rewritten as

$$G_{m-1}(z) = z G_m(z) - z k_m F_{m-1}(z) \quad (3.17)$$

Substituting Eq.(3.17) into Eq.(3.15) yields

$$F_m(z) = F_{m-1}(z) + k_m \{G_m(z) - k_m F_{m-1}(z)\} \quad (3.18)$$

Therefore, the $(m-1)^{\text{th}}$ order forward prediction error transfer function can be written in terms of the m^{th} order forward and backward prediction error transfer functions as follows:

$$F_{m-1}(z) = \frac{F_m(z) - k_m G_m(z)}{1 - k_m^2} \quad (3.19)$$

where $k_m = b_{m,m} \neq 1$. Recalling Eq.(3.13), that is,

$$G_m(z) = z^{-m} F_m(z^{-1}) \quad (3.20)$$

By substituting Eq.(3.20) into Eq.(3.19) we find that

$$F_{m-1}(z) = \frac{F_m(z) - k_m z^{-m} F_m(z^{-1})}{1 - k_m^2} \quad (3.21)$$

Thus, from Eq.(3.21) we see that the next lower order polynomial $F_{m-1}(z)$ can be calculated knowing $F_m(z)$. Following the foregoing procedure we can find $k_{m-1} = b_{m-1,m-1}$ from $F_{m-1}(z)$, and also obtain $F_{m-2}(z)$. This way, for a given M^{th} order polynomial $F_M(z)$, we can determine all the lattice reflection coefficients k_m , $m = M, M-1, \dots, 2, 1$. This procedure is known as the step-down procedure [Ref. 1], and can be summarized as follows: Let the given M^{th} order polynomial be

$$F_M(z) = \sum_{i=0}^M b_{M,i} z^{-i} \quad (3.22)$$

and by replacing M with m we have an m^{th} order the polynomial for m lattice sections

$$F_m(z) = \sum_{i=0}^m b_{m,i} z^{-i} \quad (3.23)$$

where $m = M, M-1, \dots, 2, 1$. We now define another polynomial given by

$$F_m(z^{-1}) = \sum_{i=0}^m b_{m,i} z^i \quad (3.24)$$

As we step through the procedure from $m = M$ to $m = 1$ Eq.(3.24) can be expressed as

$$F_m(z^{-1}) = \sum_{i=0}^m b_{m,m-i} z^{m-i} \quad (3.25)$$

Define $k_m = b_{m,m}$ and obtain the $m-1^{\text{th}}$ order polynomial as follows:

$$F_{m-1}(z) = \frac{F_m(z) - k_m z^{-m} F_m(z^{-1})}{1 - k_m^2} \quad (3.26)$$

Substituting Eqs.(3.23) and (3.24) into Eq.(3.26) yields

$$\sum_{i=0}^{m-1} b_{m-1,i} z^{-i} = \frac{\sum_{i=0}^m b_{m,i} z^{-i} - k_m z^{-m} \sum_{i=0}^m b_{m,m-i} z^{-i}}{1 - k_m^2}$$

$$\sum_{i=0}^{m-1} b_{m-1,i} z^{-i} = \frac{\sum_{i=0}^m b_{m,i} z^{-i} - k_m \sum_{i=0}^m b_{m,m-i} z^{-i}}{1 - k_m^2} \quad (3.27)$$

Then, by equating the coefficients of like powers of z on both sides of Eq.(3.27), we obtain the computational expression for the $(m-1)^{\text{th}}$ order polynomial coefficients as

$$b_{m-1,i} = \frac{b_{m,i} - k_m b_{m,m-i}}{1 - k_m^2} \quad (3.28)$$

where $m = M, M-1, \dots, 1$ and $i = 0, 1, \dots, m-1$, $k_m = b_{m,m}$ and $|k_m| < 1$ for a minimum phase polynomial $F_m(z)$.

D. LINEAR PHASE FIR LATTICE FILTER

In the foregoing we considered obtaining a lattice structure from a given FIR transfer function, and vice versa. In what follows we will deal with a special case of FIR filtering, namely, the linear phase FIR filter. Let $\{h(n)\}$ be a causal finite duration linear phase sequence defined over the interval $0 \leq n \leq N-1$. From Eqs.(3.2) and (3.3), the z transform of $\{h(n)\}$ can be written as

$$H_{N-1}(z) = F_M(z) + z^{-D} G_M(z) \quad (3.29)$$

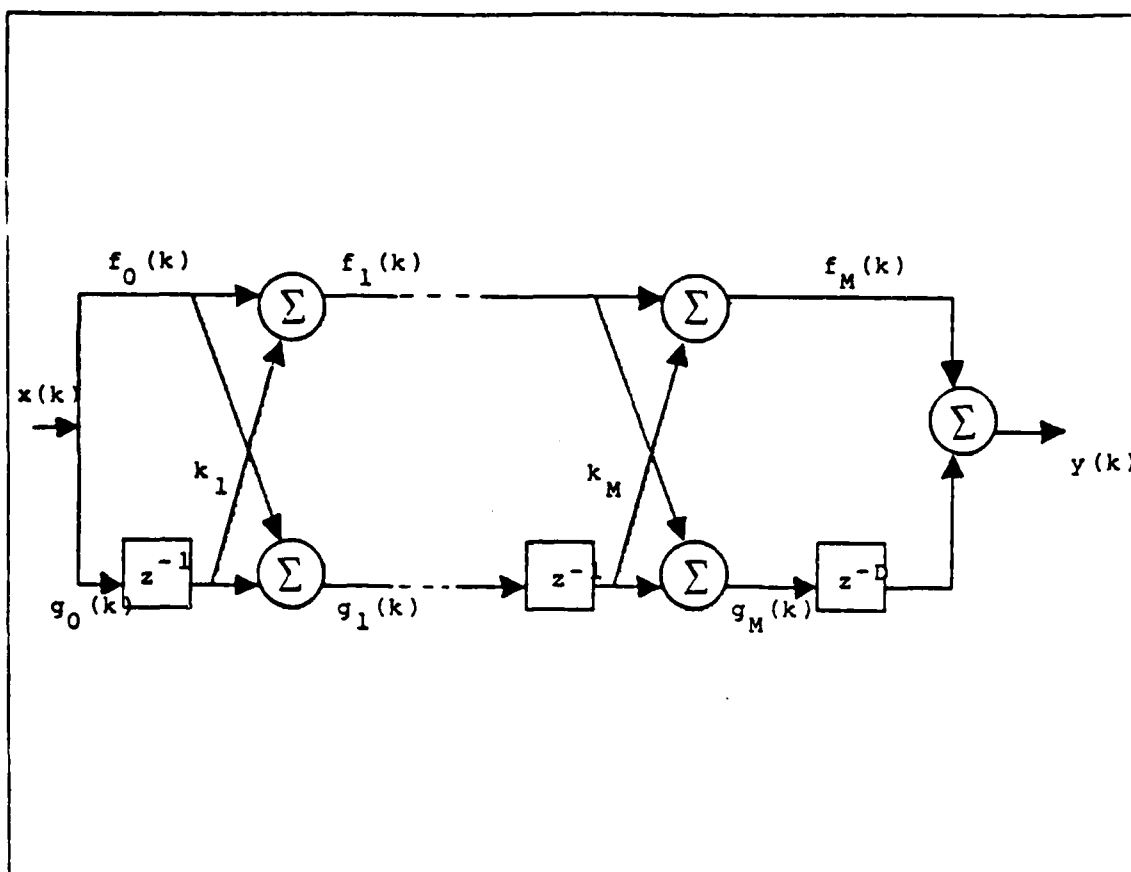


Figure 3.4 Linear Phase FIR Lattice Filter.

where $F_M(z)$ and $G_M(z)$ are forward and backward filter transfer functions, respectively, and D represents the number of unit delays. We now consider the N odd and N even cases separately.

N odd case: For N odd, Eq.(3.29) can be written as

$$\begin{aligned}
 H_{N-1}(z) &= a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{(N-3)/2} z^{-(N-3)/2} + a_{(N-1)/2} z^{-(N-1)/2} \\
 &\quad + a_{(N-3)/2} z^{-(N+1)/2} + \dots + a_2 z^{-(N-3)} + a_1 z^{-(N-2)} + a_0 z^{-(N-1)} \\
 &= a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{(N-3)/2} z^{-(N-3)/2} + 2(1/2) a_{(N-1)/2} z^{-(N-1)/2} \\
 &\quad + a_{(N-3)/2} z^{-(N+1)/2} + \dots + a_2 z^{-(N-3)} + a_1 z^{-(N-2)} + a_0 z^{-(N-1)}
 \end{aligned}$$

Note that we are splitting the coefficient $a_{(N-1)/2}$ into two coefficients of value $(1/2)a_{(N-1)/2}$ each.

$$\begin{aligned}
 H_{N-1}(z) = & a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{(N-3)/2} z^{-(N-3)/2} \\
 & + (1/2) a_{(N-1)/2} z^{-(N-1)/2} + z^{-(N-1)/2} [(1/2) a_{(N-1)/2} \\
 & + a_{(N-3)/2} z^{-1} + \dots + a_2 z^{-(N-5)/2} + a_1 z^{-(N-3)/2} \\
 & + a_0 z^{-(N-1)/2}]
 \end{aligned} \tag{3.30}$$

$$H_{N-1}(z) = F_M(z) + z^{-(N-1)/2} G_M(z) \tag{3.31}$$

where the order of the polynomials $F_M(z)$ and $G_M(z)$, $M = (N-1)/2$, and the number of unit delays, $D = (N-1)/2$, the forward transfer function,

$$F_M(z) = a_0 + a_1 z^{-1} + \dots + a_{(N-3)/2} z^{-(N-3)/2} + (1/2) a_{(N-1)/2} z^{-(N-1)/2} \tag{3.32}$$

$$F_M(z^{-1}) = a_0 + a_1 z + \dots + a_{(N-3)/2} z^{(N-3)/2} + (1/2) a_{(N-1)/2} z^{(N-1)/2} \tag{3.33}$$

and the backward transfer function,

$$\begin{aligned}
 G_M(z) = & z^{-(N-1)/2} F_M(z^{-1}) = (1/2) a_{(N-1)/2} - a_{(N-3)/2} z^{-1} \\
 & + \dots + a_1 z^{-(N-3)/2} + a_0 z^{-(N-1)/2}
 \end{aligned} \tag{3.34}$$

N even case: For N even, Eq.(3.29) can be written as

$$H_{N-1}(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{(N-2)/2} z^{-(N-2)/2} + a_{(N-2)/2} z^{-N/2} \\ + \dots + a_2 z^{-(N-3)} + a_1 z^{-(N-2)} + a_0 z^{-(N-1)}$$

$$H_{N-1}(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{(N-2)/2} z^{-(N-2)/2} \quad (3.35) \\ + z^{-N/2} [a_{(N-2)/2} + \dots + a_2 z^{-(N-6)/2} \\ + a_1 z^{-(N-4)/2} + a_0 z^{-(N-2)/2}]$$

$$H_{N-1}(z) = F_M(z) + z^{-N/2} G_M(z) \quad (3.36)$$

where the order of the polynomials $F_M(z)$ and $G_M(z)$, $M = \{(N/2)-1\}$, and the number of unit delays, $D = N/2$, the forward transfer function,

$$F_M(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_{(N-2)/2} z^{-(N-2)/2} \quad (3.37)$$

$$F_M(z^{-1}) = a_0 + a_1 z + a_2 z^2 + \dots + a_{(N-2)/2} z^{(N-2)/2} \quad (3.38)$$

and the backward transfer function,

$$G_M(z) = z^{-(N-2)/2} F_M(z^{-1}) = a_{(N-2)/2} + \dots + a_2 z^{-(N-6)/2} \quad (3.39) \\ + \dots + a_1 z^{-(N-4)/2} + a_0 z^{-(N-2)/2}$$

Figure 3.4 shows a linear phase FIR lattice realization where the filter output is obtained by adding the M^{th} order forward prediction error, and the M^{th} order delayed (by D unit delays) backward prediction error. Combining the discussion in this section and that in Section (III.C), we can summarize the algorithm for converting a given FIR transfer function into a lattice structure as follows:

- (i) Find if N is even, or odd
- (ii) For N odd: $M = (N-1)/2$, $D = (N-1)/2$
- (iii) For N even: $M = \{(N/2)-1\}$, $D = N/2$
- (iv) From $F_M(z)$, obtain M reflection coefficients as discussed in Eqs.(3.22) to (3.26)
- (v) Implement the lattice as shown in Figure 3.4

E. LATTICE REALIZATION OF A GENERAL FIR TRANSFER FUNCTION

In the foregoing we have considered a polynomial $F_M(z)$ with $b_{M,0} = 1$. However, in general we have $b_{M,0} \neq 1$. In this section, we shall modify the lattice realization algorithm presented in Section (III.C) to suit an arbitrary FIR transfer function with $b_{M,0} \neq 1$ [Ref. 22]. The m^{th} order polynomials $F_m(z)$ and $G_m(z)$ can be obtained from Eqs.(3.15) and (3.16):

$$F_m(z) = s_m F_{m-1}(z) + k_m z^{-1} G_{m-1}(z) \quad (3.40)$$

$$G_m(z) = k_m F_{m-1}(z) + s_m z^{-1} G_{m-1}(z) \quad (3.41)$$

where the coefficients $s_m = b_{m,0}$ and $k_m = b_{m,m}$ are recognized as the reflection coefficients. Eq.(3.41) can be rewritten as

$$G_{m-1}(z) = s_m^{-1} z G_m(z) - s_m^{-1} k_m z F_{m-1}(z) \quad (3.42)$$

Substituting Eq.(3.42) into Eq.(3.40) yields

$$F_m(z) = s_m F_{m-1}(z) + s_m^{-1} k_m \{G_m(z) - k_m F_{m-1}(z)\} \quad (3.43)$$

Therefore, the $(m-1)^{\text{th}}$ order forward prediction error transfer function can be written in terms of the m^{th} order forward and backward prediction error transfer functions as follows:

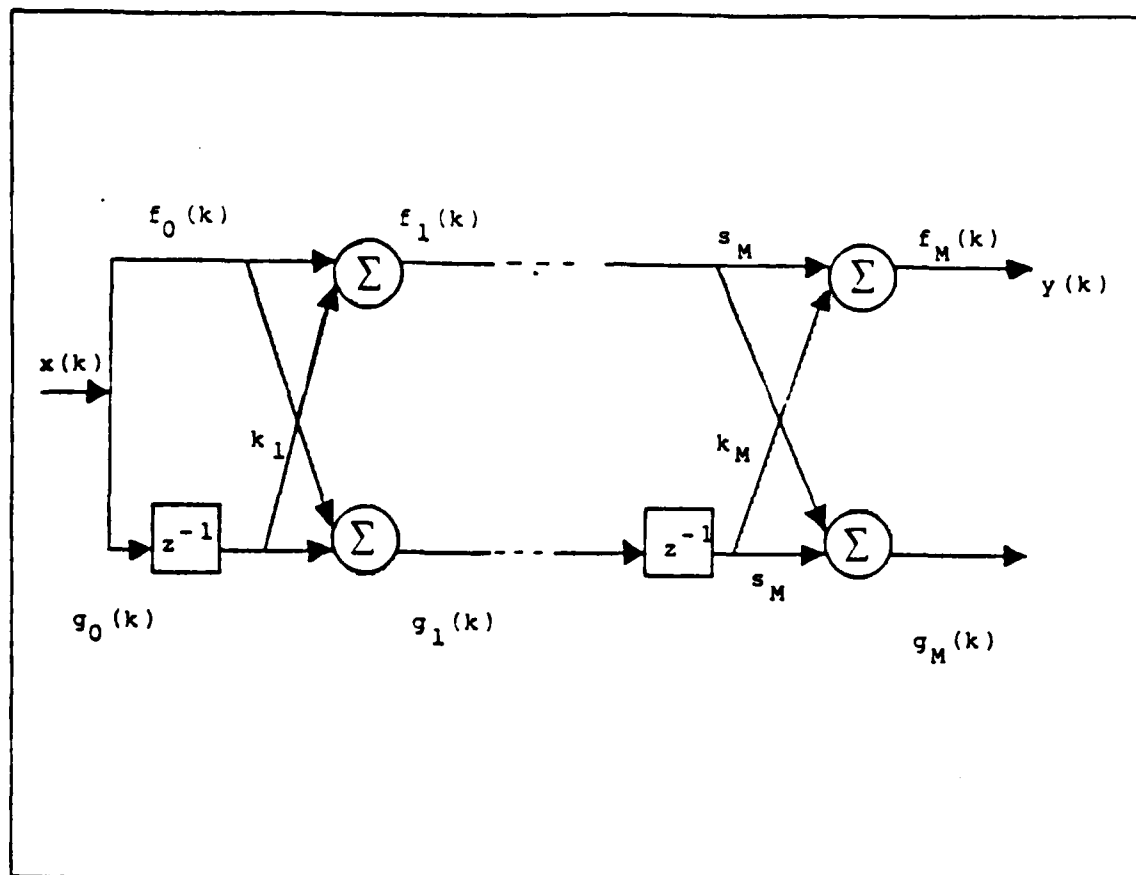


Figure 3.5 FIR Lattice.

$$F_{m-1}(z) = \frac{s_m F_m(z) - k_m G_m(z)}{s_m^2 - k_m^2} \quad (3.44)$$

where $s_m \neq k_m$. Substituting Eqs.(3.20), (3.23) and (3.25) into Eq.(3.44) yields

$$\sum_{i=0}^{m-1} b_{m-1,i} z^{-i} = \frac{s_m \sum_{i=0}^m b_{m,i} z^{-i} - k_m z^{-m} \sum_{i=0}^m b_{m,m-i} z^{-i}}{s_m^2 - k_m^2}$$

$$\sum_{i=0}^{m-1} b_{m-1,i} z^{-i} = \frac{s_m \sum_{i=0}^m b_{m,i} z^{-i} - k_m \sum_{i=0}^m b_{m,m-i} z^{-i}}{s_m^2 - k_m^2} \quad (3.45)$$

From Eq.(3.45), we obtain the computational expression for the $(m-1)^{\text{th}}$ order polynomial coefficients is

$$b_{m-1,i} = \frac{s_m b_{m,i} - k_m b_{m,m-i}}{s_m^2 - k_m^2} \quad (3.46)$$

where $m = M, M-1, \dots, 1$ and $i = 0, 1, \dots, m-1$, $k_m = b_{m,m}$, $s_m = b_{m,0}$ and $s_m > k_m$ for a minimum phase polynomial $F_m(z)$. It may be noted that $s_m = 1$ for $m = 1, 2, \dots, M-1$. This indicates that we have only one s -coefficient, i.e., s_M , which requires an extra multiplication in the M^{th} lattice section as shown in Figure 3.5.

IV. LMS ALGORITHM

A. INTRODUCTION

Conventionally an adaptive filter is composed of a tapped delay line or transversal structure with adjustable coefficients or weights and an adaptive algorithm which updates the coefficients continuously based on some performance criterion.

The design of a fixed coefficient filter is based on the prior knowledge of both signal and noise. Adaptive filters, on the other hand, have the ability to adjust their own parameters automatically, and their design requires little, or no *a priori* knowledge of signal or noise characteristics [Ref. 14]. However, the designer has to choose the order of the filter and the type of the algorithm. Also the adaptive filter usually requires a large initial transient time (i.e., the initial filter convergence period).

For stationary stochastic inputs, the mean square error, the difference between the filter output and an externally supplied input called the "desired response", is a quadratic function of the filter coefficients, a paraboloid with a single fixed minimum point that can be sought by gradient techniques [Ref. 16].

In the previous chapter we showed that the operation of a multistage lattice filter is completely described by specifying the sequence of reflection coefficients that characterize the individual stages of the filter. In this chapter we briefly discuss the least-mean-square (LMS) adaptive algorithm that results from attempting to minimize the mean square error and present a summary of some LMS algorithms for lattice that have been reported in the literature. Also included are the computer simulation results.

The least-mean-square (LMS) adaptive algorithm minimizes the mean square error $\epsilon(k)$ by recursively altering the filter coefficient vector $\underline{E}(k)$ at each sampling instant. The original Widrow-Hoff LMS algorithm is $\underline{E}(k+1) = \underline{E}(k) + 2\mu\epsilon(k)\underline{X}(k)$, where μ is a convergence factor controlling stability and rate of adaptation [Ref. 15]. The algorithm is based on the method of steepest descent, moving $\underline{E}(k)$ in proportion to the instantaneous gradient estimate of the mean square error.

When the filter input is stationary, the backward prediction errors are orthogonal to each other, with the result that the successive stages of the lattice filter are decoupled from each other [Ref 8]. This means that the global optimization of a multistage lattice filter may indeed be accomplished as a sequence of local optimization

problems, one at each stage of the lattice filter. Accordingly, it is a straightforward matter to increase the order of the lattice filter by simply adding one or more stages without affecting the earlier design computations. The successive orthogonalization provided by the lattice offers advantages in adaptive convergence rate which cannot be achieved with tapped delay lines [Ref. 17,18].

B. . SUMMARY OF THE LMS ALGORITHM

The LMS algorithm uses an estimate of the gradient of the mean square error obtained from the adaptive linear combiner which is a combination of a transversal structure and an adder. The adaptive linear combiner can be shown in two basic ways, depending on whether the input is available in parallel form (multiple inputs), or in serial form (single input) [Ref. 6].

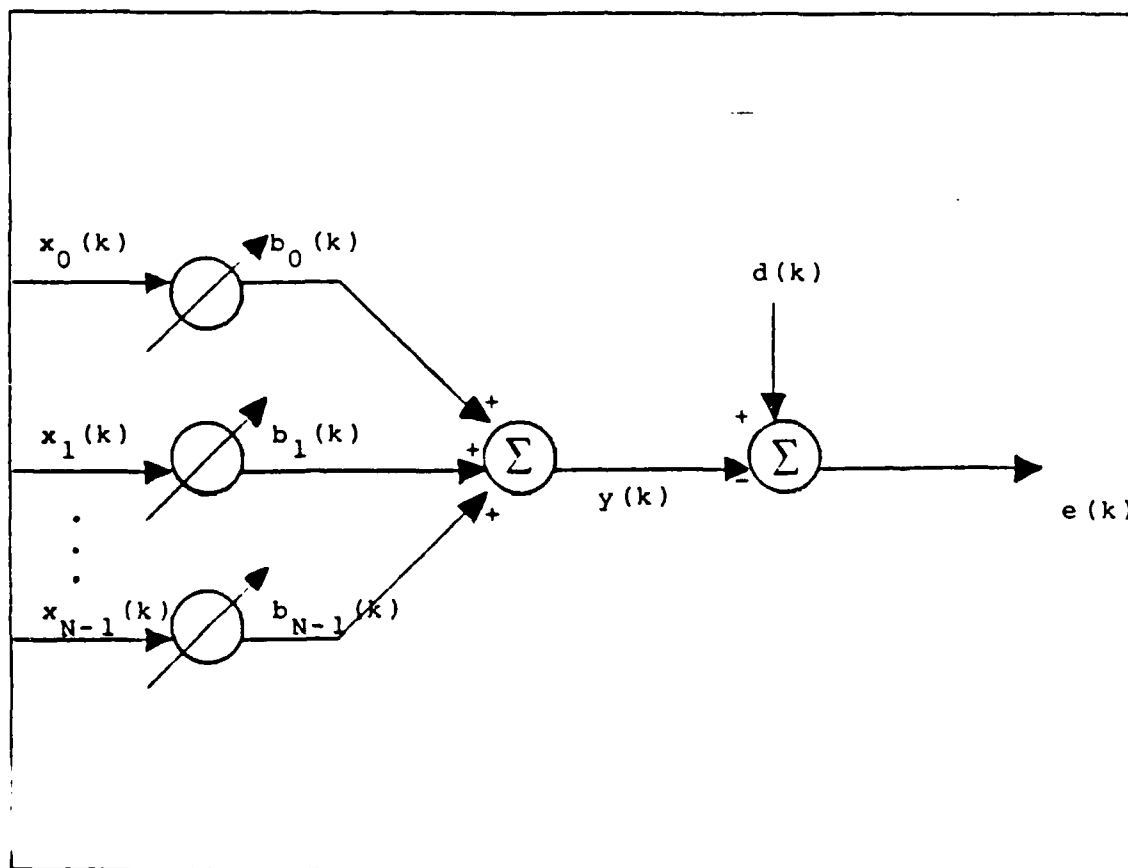


Figure 4.1 Parallel Form.

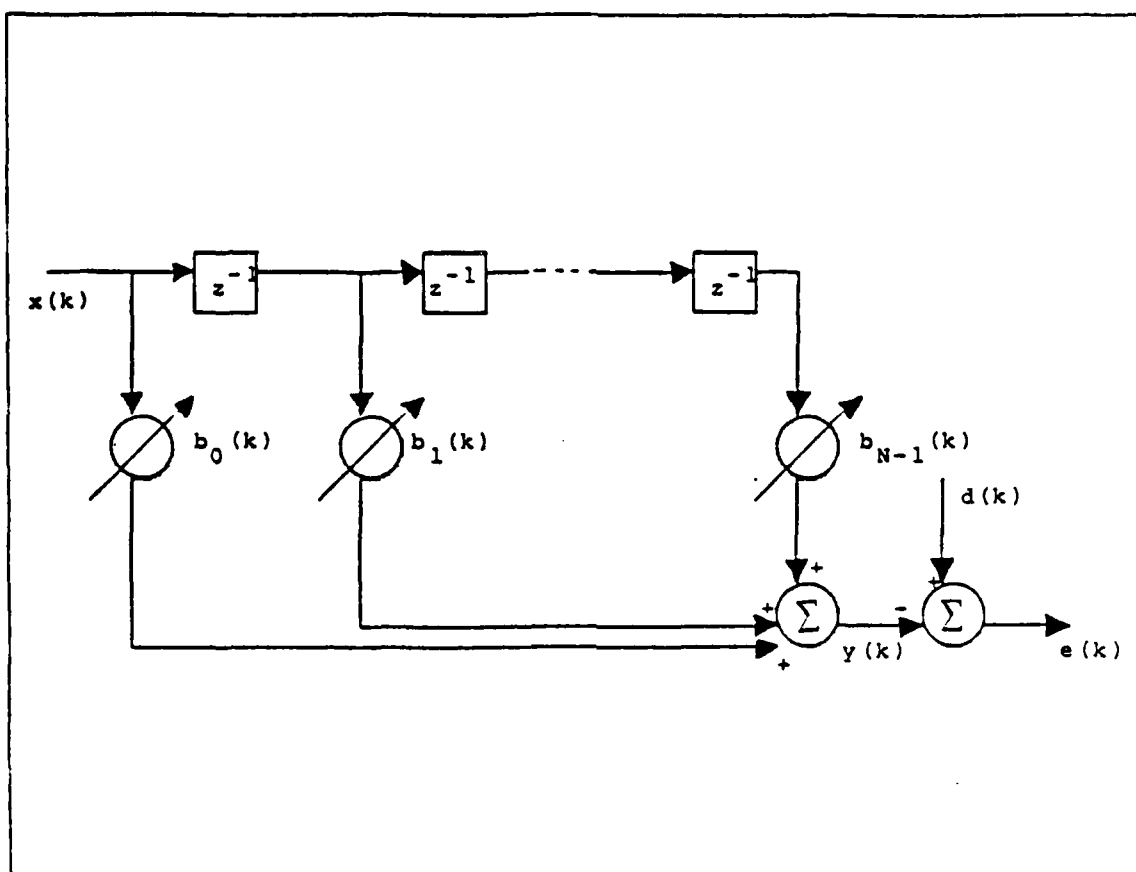


Figure 4.2 Serial Form.

In the following we present a brief derivation of the LMS algorithm. Let us choose the single input form, then the filter output is given by

$$\begin{aligned}
 y(k) &= b_0(k)x(k) + b_1(k)x(k-1) + \dots + b_{N-1}(k)x(k-N+1) \\
 &= \sum_{i=0}^{N-1} b_i(k) x(k-i) \\
 &= \underline{x}^T(k) \underline{g}(k) \\
 &= \underline{g}^T(k) \underline{x}(k)
 \end{aligned} \tag{4.1}$$

where \underline{x} and \underline{g} are the input signal vector and the filter coefficient vector, respectively and $(N-1)$ is the order of the filter. The input signal vector \underline{x} and the filter coefficient vector \underline{g} are defined as

$$\underline{X}(k) = \begin{bmatrix} x(k) \\ x(k-1) \\ \vdots \\ x(k-N+1) \end{bmatrix} \quad \underline{B}(k) = \begin{bmatrix} b_0(k) \\ b_1(k) \\ \vdots \\ b_{N-1}(k) \end{bmatrix}$$

Therefore, the output $y(k)$ is equal to the inner product of $\underline{X}(k)$ and $\underline{B}(k)$.

The error $e(k)$ is defined as the difference between the desired response $d(k)$ and the actual response $y(k)$,

$$e(k) = d(k) - \underline{X}^T(k)\underline{B}(k) = d(k) - \underline{B}^T(k)\underline{X}(k) \quad (4.2)$$

The purpose of the adaptive algorithm is to adjust the filter coefficients of the adaptive linear combiner to minimize the mean-square error. A general expression for mean square error as a function of the filter coefficient values, assuming that the input signals and the desired response are statistically stationary and that the filter coefficients are fixed, can be derived in the following manner [Ref 6]. The squared error is

$$e^2(k) = d^2(k) - 2d(k)\underline{X}^T(k)\underline{B}(k) + \underline{B}^T(k)\underline{X}(k)\underline{X}^T(k)\underline{B}(k) \quad (4.3)$$

Taking the expected value of both sides yields the mean square error,

$$E[e^2(k)] = E[d^2(k)] - 2 E[d(k)\underline{X}^T(k)] \underline{B}(k) + \underline{B}^T(k) E[\underline{X}(k)\underline{X}^T(k)] \underline{B} \quad (4.4)$$

Defining the vector \underline{P} as the cross-correlation between the desired response and the input vector, we have

$$\underline{P} = E [d(k) \underline{X}(k)] \quad (4.5)$$

Similarly the input autocorrelation matrix \underline{R} is defined as

$$\underline{R} = E [\underline{X}(k) \underline{X}^T(k)] \quad (4.6)$$

Thus the mean-square error can be expressed as

$$\epsilon(k) = E[e^2(k)] = E[d^2(k)] - 2 \underline{P}^T \underline{B}(k) + \underline{B}^T(k) \underline{R} \underline{B}(k) \quad (4.7)$$

The gradient $\nabla \epsilon(k)$ of the mean square error function is obtained by differentiating Eq.(4.7) with respect to the filter coefficient vector as follows:

$$\nabla \epsilon(k) = \begin{bmatrix} \frac{\partial E[e^2(k)]}{\partial b_0(k)} \\ \vdots \\ \frac{\partial E[e^2(k)]}{\partial b_{N-1}(k)} \end{bmatrix} = -2 \underline{P} + 2 \underline{R} \underline{B}(k) \quad (4.8)$$

The LMS algorithm is an implementation of the method of steepest descent. According to this method, the next filter coefficient vector is equal to the present filter coefficient vector $\underline{B}(k)$ plus a change proportional to negative of the gradient, $\nabla \epsilon(k)$:

$$\underline{B}(k+1) = \underline{B}(k) - \mu \nabla \epsilon(k) \quad (4.9)$$

The parameter μ is the factor that controls stability and rate of convergence. In other words, the first term on the right hand side consists of the past information and the second term represents the new, or updated information.

The LMS algorithm estimates an instantaneous gradient in a crude but efficient manner by assuming that $e^2(k)$, the square of a single error sample, is an estimate of the mean-square error and by differentiating $e^2(k)$ with respect to $\underline{B}(k)$. The estimated gradient is given by the following expression

$$\hat{\nabla} \varepsilon(k) = \begin{bmatrix} \frac{\partial e^2(k)}{\partial b_0(k)} \\ \vdots \\ \frac{\partial e^2(k)}{\partial b_{N-1}(k)} \end{bmatrix} = 2 e(k) \begin{bmatrix} \frac{\partial e(k)}{\partial b_0(k)} \\ \vdots \\ \frac{\partial e(k)}{\partial b_{N-1}(k)} \end{bmatrix} \quad (4.10)$$

The estimated gradient components are related to the partial derivatives of the instantaneous error with respect to the filter coefficient components. Thus the expression for the gradient estimate can be simplified to

$$\hat{\nabla} \varepsilon(k) = -2 e(k) \underline{X}(k) \quad (4.11)$$

Using this estimate in place of the true gradient in Eq.(4.11) yields the Widrow-Hoff LMS algorithm

$$\underline{B}(k+1) = \underline{B}(k) + 2 \mu e(k) \underline{X}(k) \quad (4.12)$$

Since the filter coefficient changes at each iteration are based on imperfect gradient estimates, we would expect the adaptive process to be noisy, that is, it would not follow the true line of steepest descent on the performance surface. The LMS algorithm can be implemented in a practical system without squaring, averaging, or differentiation and is elegant in its simplicity and efficiency. Each component of the gradient vector is obtained from a single data sample without perturbing the filter coefficient vector.

C. ADAPTIVE LATTICE ALGORITHMS

The parameters to update in a multistage lattice are its reflection coefficients. Several algorithms have been proposed in the literature for updating the lattice reflection coefficients [Ref. 7,8,17,18,20,23-28]. In this section we briefly summarize some of those algorithms. Consider a lattice filter of order M . For stage m of the filter, the flow of signals is described by the following pair of equations.

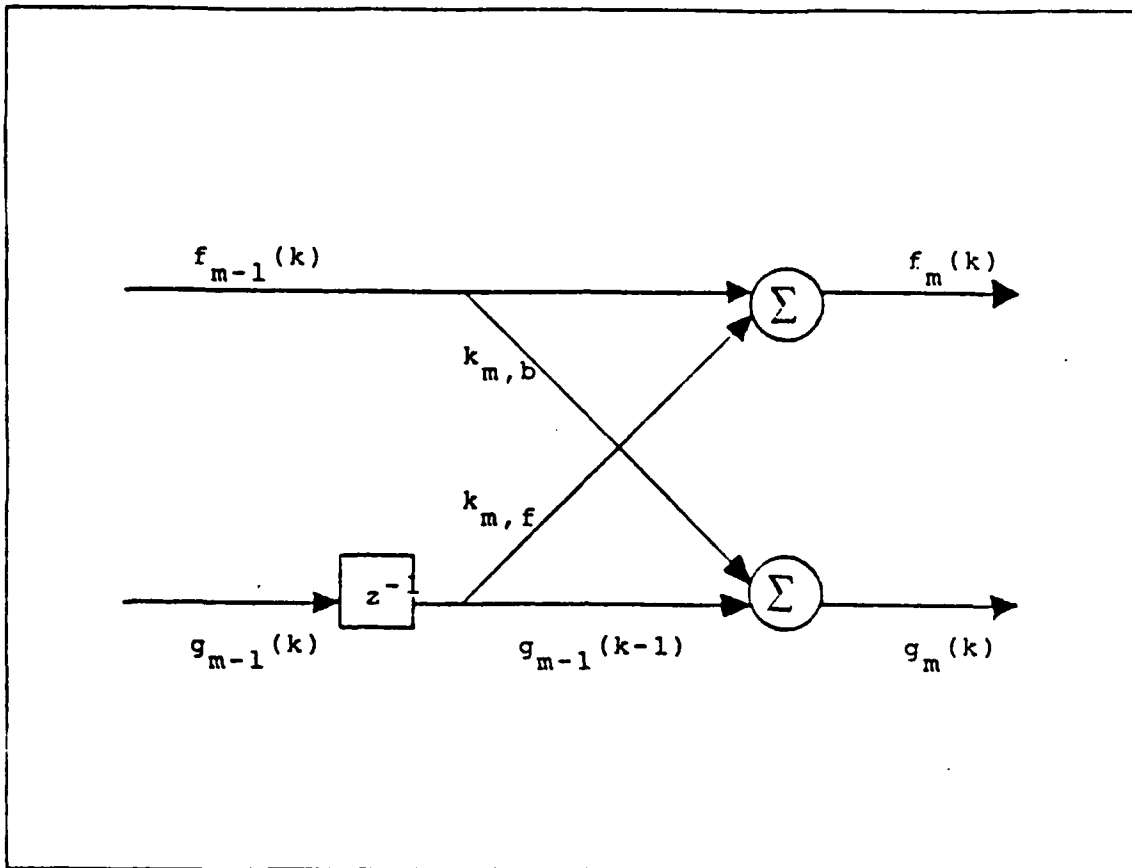


Figure 4.3 A Single Stage of Lattice Filter.

$$\begin{aligned} f_m(k) &= f_{m-1}(k) + k_{m,f} g_{m-1}(k-1) \\ g_m(k) &= g_{m-1}(k-1) + k_{m,b} f_{m-1}(k) \end{aligned} \quad (4.13)$$

where $m = 1, 2, \dots, M$

For stage m , the forward reflection coefficient is denoted by $k_{m,f}$ and the backward reflection coefficient is denoted by $k_{m,b}$. $f_m(k)$ and $g_m(k)$ are the forward prediction error and backward prediction error of stage m respectively. Before presenting the adaptive algorithms for the lattice, let us quickly summarize some non-adaptive reflection coefficient estimation methods. Later on we can obtain the adaptive updating equations as approximations to these methods.

1. Non-adaptive Methods

When the lattice coefficients have fixed, non-adaptive values, several methods have been proposed for computing these values as functions of the correlation statistics of the input. Two of these which are based on mean-square error (MSE) minimization [Ref.28] are given below.

Method 1 : In this method, we choose $k_{m,f}$ to minimize the mean forward prediction error power, $E[f_m^2(k)]$ and $k_{m,b}$ to minimize the mean backward prediction error power, $E[g_m^2(k)]$. Here we give the final equations for $k_{m,f}$ and $k_{m,b}$ without going into the details. The forward and backward reflection coefficients are given by

$$\begin{aligned} k_{m,f} &= \frac{E[f_{m-1}(k)g_{m-1}(k-1)]}{E[g_{m-1}^2(k-1)]} \\ k_{m,b} &= \frac{E[f_{m-1}(k)g_{m-1}(k-1)]}{E[f_{m-1}^2(k)]} \end{aligned} \quad (4.14)$$

where both $k_{m,f}$ and $k_{m,b}$ are obtained from the cross-correlations between the $(m-1)^{th}$ order forward and backward prediction errors normalized by respective prediction error powers.

Method 2 : For a single channel lattice using real data and coefficients we can, however, show that $k_{m,f} = k_{m,b} = k_m$. Based on the condition that $k_{m,f} = k_{m,b} = k_m$, we can now minimize either $E[f_m^2(k)]$, or $E[g_m^2(k)]$ in order to obtain an optimum k_m . However, it seems more logical to minimize the sum $E[f_m^2(k)] + E[g_m^2(k)]$ as suggested in [Ref. 28]. The resulting reflection coefficient equation in its final form is given by

$$k_{m,f} = \frac{2 E[f_{m-1}(k)g_{m-1}(k-1)]}{E[f_{m-1}^2(k)] + E[g_{m-1}^2(k-1)]} \quad (4.15)$$

where we have normalized the cross-correlation term with respect to the sum of mean prediction error powers. In both of these methods, the coefficients at stage m can be computed independently of those following that stage. Thus, optimum values can be

computed successively along the structure without affecting the other coefficients. This phenomenon lends to the modular nature of the lattice structure.

2. Adaptive Methods

An instantaneous, gradient-descent adaptive algorithm minimizes a mean-square error criterion can be derived for a generalized problem. An adaptive lattice structure is suggested by incorporating time varying coefficients $k_{m,f}(k)$ and $k_{m,b}(k)$ in the lattice and generating algorithms for the two methods described above. The resulting procedures are :

Method 1 : Corresponding to method 1 of non-adaptive procedures metioned above, we can obtain the following update equations

$$\begin{aligned} k_{m,f}(k+1) &= k_{m,f}(k) + \frac{\mu}{\sigma_{m-1,f}^2(k)} [f_m(k) g_{m-1}(k-1)] \\ k_{m,b}(k+1) &= k_{m,b}(k) + \frac{\mu}{\sigma_{m-1,g}^2(k)} [f_{m-1}(k) g_m(k)] \end{aligned} \quad (4.16)$$

where μ is an adaptive step size parameter, λ is a positive weighting constant, and the forward power estimate at the $(m-1)^{th}$ stage, $\sigma_{m-1,f}^2(k)$, is

$$\sigma_{m-1,f}^2(k) = \lambda \sigma_{m-1,f}^2(k-1) + (1-\lambda) [g_{m-1}^2(k-1)]$$

and the backward power estimate at the $(m-1)^{th}$ stage, $\sigma_{m-1,b}^2(k)$, is

$$\sigma_{m-1,g}^2(k) = \lambda \sigma_{m-1,g}^2(k-1) + (1-\lambda) [f_{m-1}^2(k)]$$

Method 2 : Eq.(4.15) can be approximated to obtain a recursive update equation as follows:

$$k_m(k+1) = k_m(k) + \mu [f_m(k)g_{m-1}(k-1) + f_{m-1}(k)g_m(k)] \quad (4.17)$$

For the basic structure of the single-channel lattice, ie, $k_{m,f} = k_{m,b} = k_m$, the reflection coefficients or filter coefficients $k_m(k)$ may be updated using a modification of the LMS algorithm of the LMS algorithm [Ref. 17,20].

$$k_m(k+1) = k_m(k) + \frac{\mu}{\sigma_{m-1}^2(k)} [f_m(k)g_{m-1}(k-1) + f_{m-1}(k)g_m(k)] \quad (4.18)$$

where $\sigma_{m-1}^2(k)$ is the power estimate at the $(m-1)^{th}$ stage. Now the updated power estimate is

$$\sigma_{m-1}^2(k) = \lambda \sigma_{m-1}^2(k-1) + (1-\lambda) [g_{m-1}^2(k-1) + f_{m-1}^2(k)] \quad (4.19)$$

where λ is a positive weighting constant satisfying the criterion $0 < \lambda \leq 1$ then controls the bandwidth of the filter and the resulting power averaging time. A power estimate is required at each stage in the lattice due to the fact that the forward and reverse error sequences have decreased power with increased stage number.

Method 3 : A third successful method of implementing an adaptive algorithm for FIR lattice structures has been reported by Griffiths [Ref. 17,18]. This algorithm has been originally discussed for a noise canceller application. The form of the lattice noise-cancelling adaptive filter is shown in Figure 4.4. The lattice noise-cancelling adaptive filter consists of an M stage linear prediction lattice for the reference signal $x_r(k)$ together with a set of tap coefficients $V_m(k)$ which provide the noise-cancelling subtraction paths. Griffiths' algorithm is briefly presented in the following. The update equations for $k_m(k)$ are given by

$$k_m(k+1) = k_m(k) + \frac{\mu}{\sigma_{m-1}^2(k)} [f_m(k)g_{m-1}(k-1) + g_m(k)f_{m-1}(k)] \quad (4.20)$$

where the power estimate at the $(m-1)^{th}$ stage, $\sigma_{m-1}^2(k)$ is

$$\sigma_{m-1}^2(k) = \lambda \sigma_{m-1}^2(k-1) + (1-\lambda) [g_{m-1}^2(k-1) + f_{m-1}^2(k)] \quad (4.21)$$

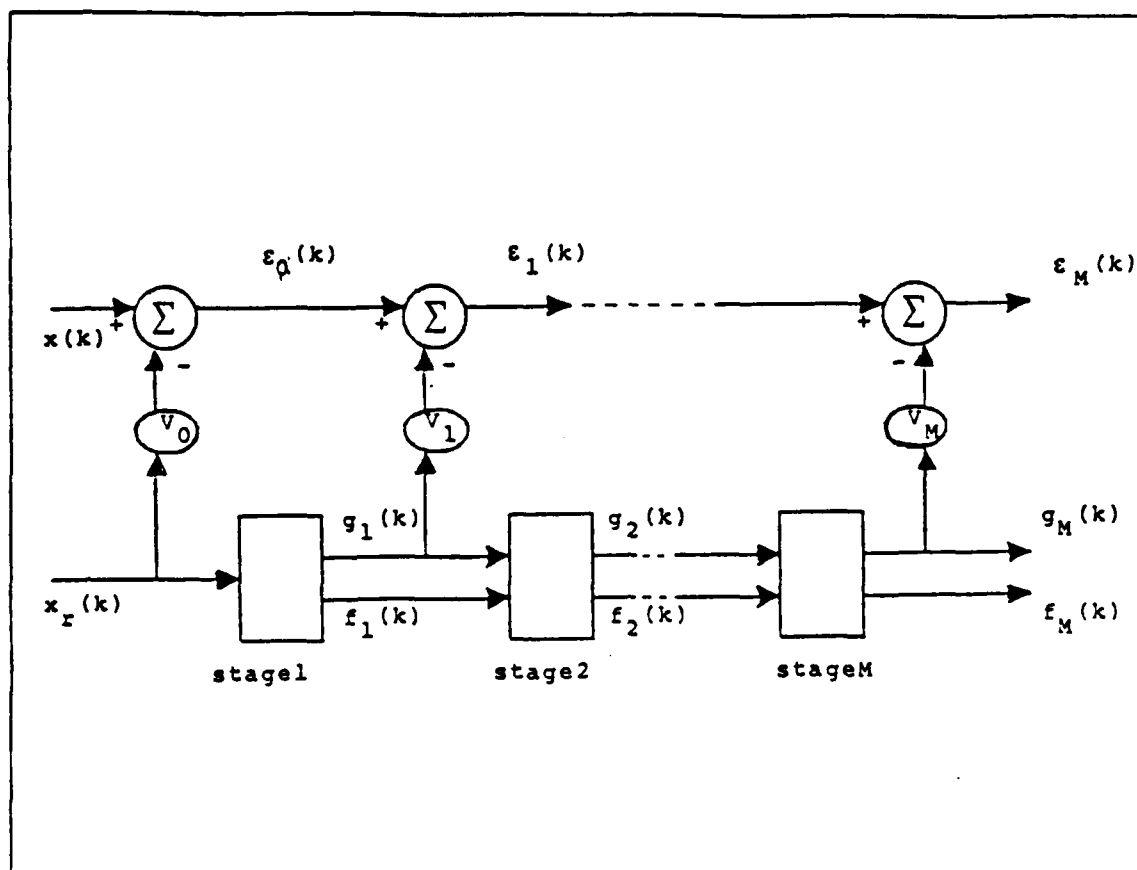


Figure 4.4 Lattice Form Implementation of Noise-cancelling Filter.

Each coefficient $V_n(k)$ can be determined independently of $V_n(k)$ for $n > m$, because of orthogonalization provided by the lattice. Thus the resulting algorithm is

$$V_m(k+1) = V_m(k) + \frac{\mu}{\gamma_m^2(k)} [\epsilon_m(k) g_m(k)] \quad (4.22)$$

where associated power measurement $\gamma_m^2(k)$ is

$$\gamma_m^2(k) = \lambda \gamma_m^2(k-1) + (1-\lambda) [g_m(k-1) g_m(k-1)] \quad (4.23)$$

and $\epsilon_m(k)$ is the m^{th} stage error signal as shown in Figure 4.4.

D. SIMULATION RESULTS

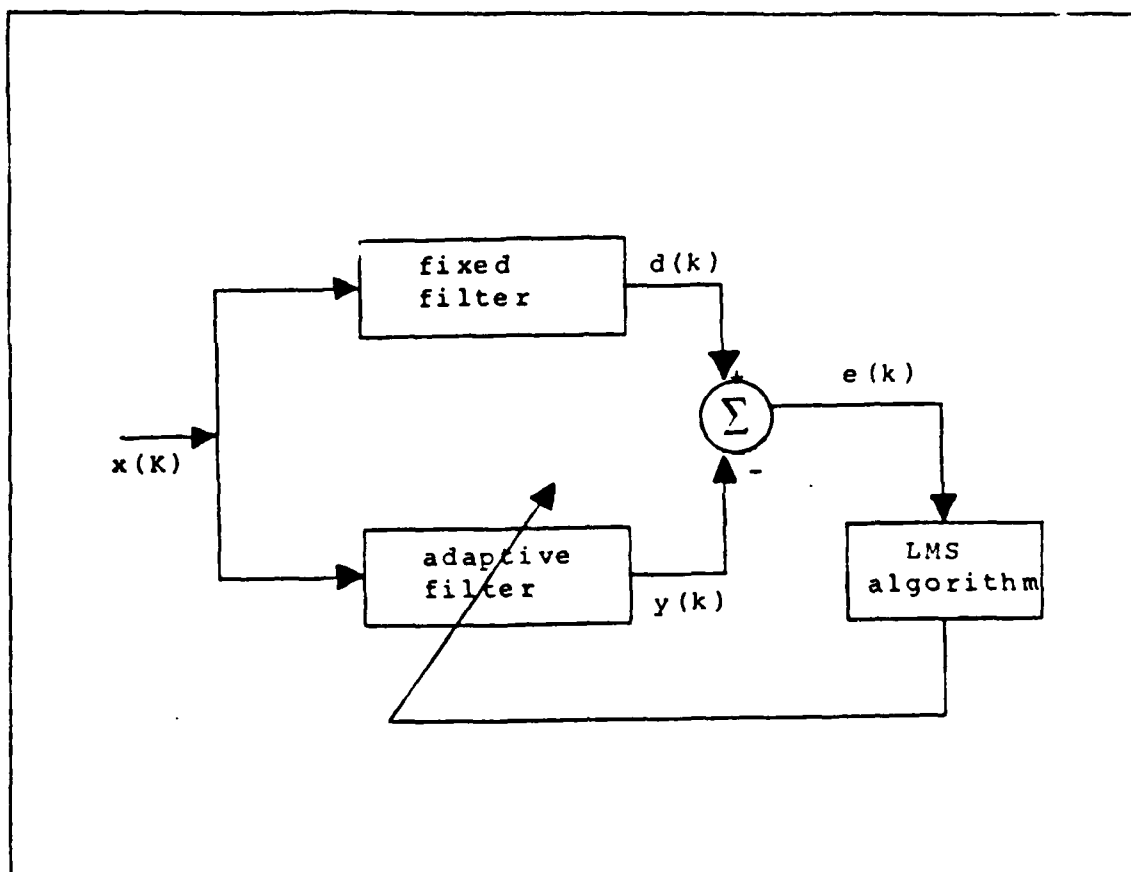


Figure 4.5 System Identification Experiment.

For the purpose of computer simulation, let us consider an approximate algorithm given by

$$k_m(k+1) = k_m(k) + \frac{\mu}{\sigma_m^2(k)} [g_{m-1}(k-1)] e(k) \quad (4.24)$$

where

$$\sigma_m^2(k) = \lambda \sigma_m^2(k-1) + (1-\lambda) g_{m-1}^2(k-1) \quad (4.25)$$

The configuration used in the simulation is a system identification experiment as shown in Figure 4.5. The fixed filter transfer function considered is given by

$$H(z) = 1 - 0.89z^{-1} + 0.25z^{-2}$$

The convergence performance of the LMS algorithm (as given by Eq.(4.24)) can be observed by plotting the error, $e(k)$, versus the update iteration, k , called learning curves. The input $x(k)$ to both fixed and adaptive filters is a white noise sequence with zero mean and unit variance. Figures 4.6 to 4.8 show the learning curves for the above example where we have used three different values for the adaptation constant, μ . In the next chapter, we derive a new algorithm for updating the reflection coefficients, based on the least mean square principle and the steepest descent algorithm. Improvement in convergence speed will be shown using the new algorithm.

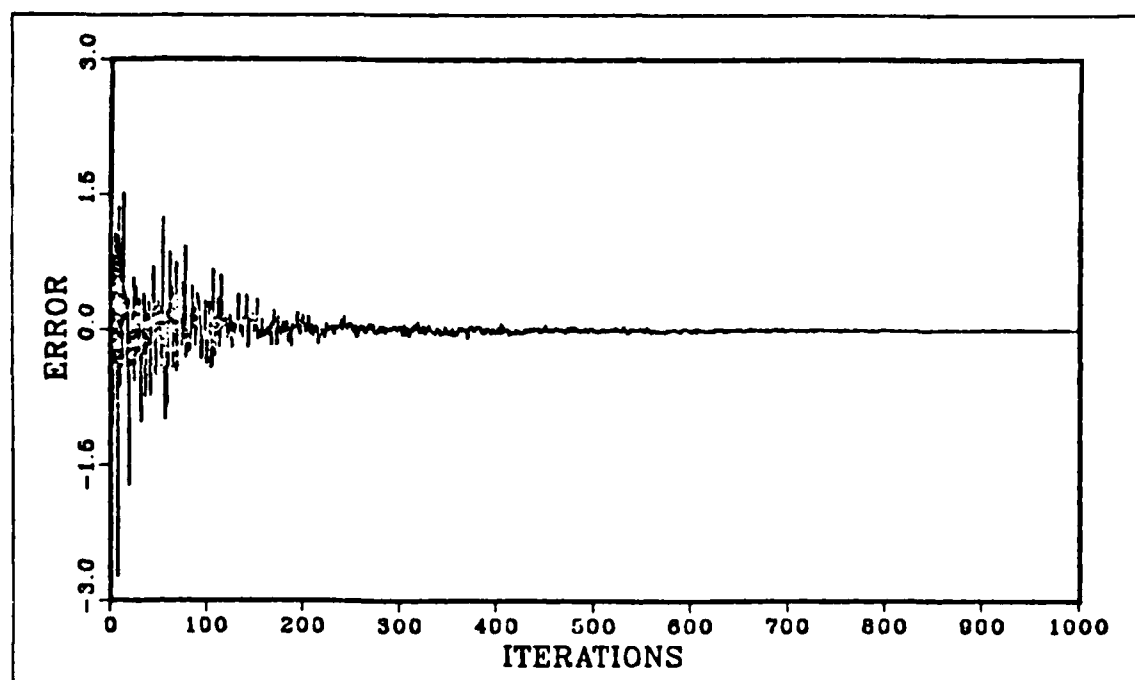


Figure 4.6 Learning Curve ($\mu = 0.01$).

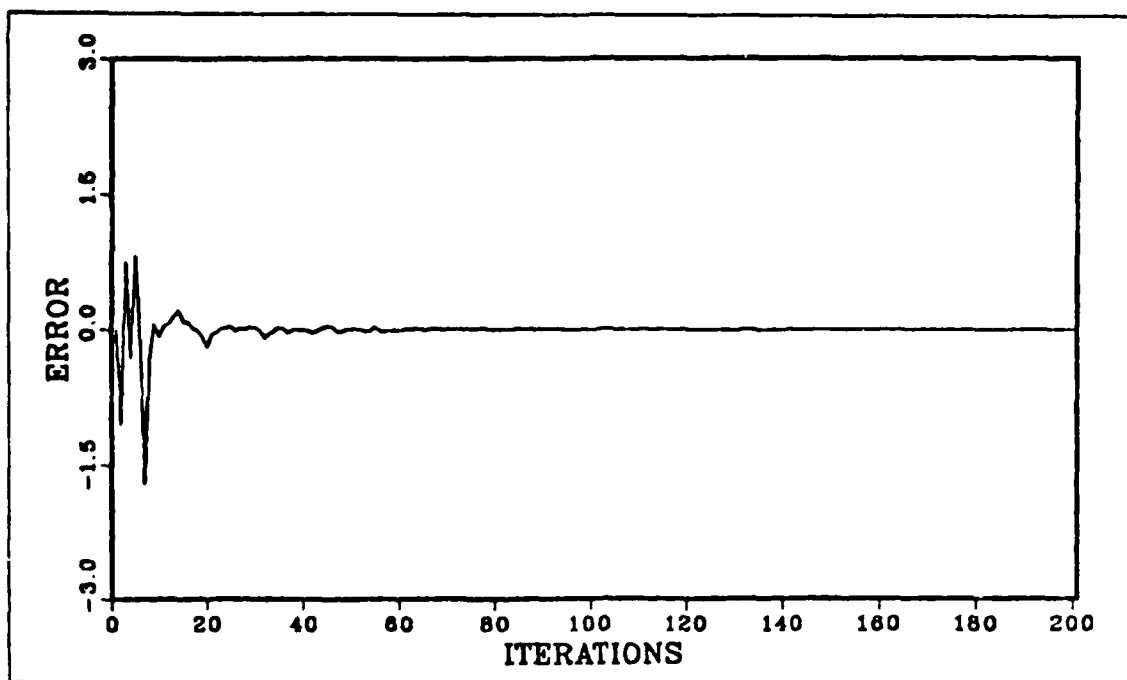


Figure 4.7 Learning Curve ($\mu = 0.1$).

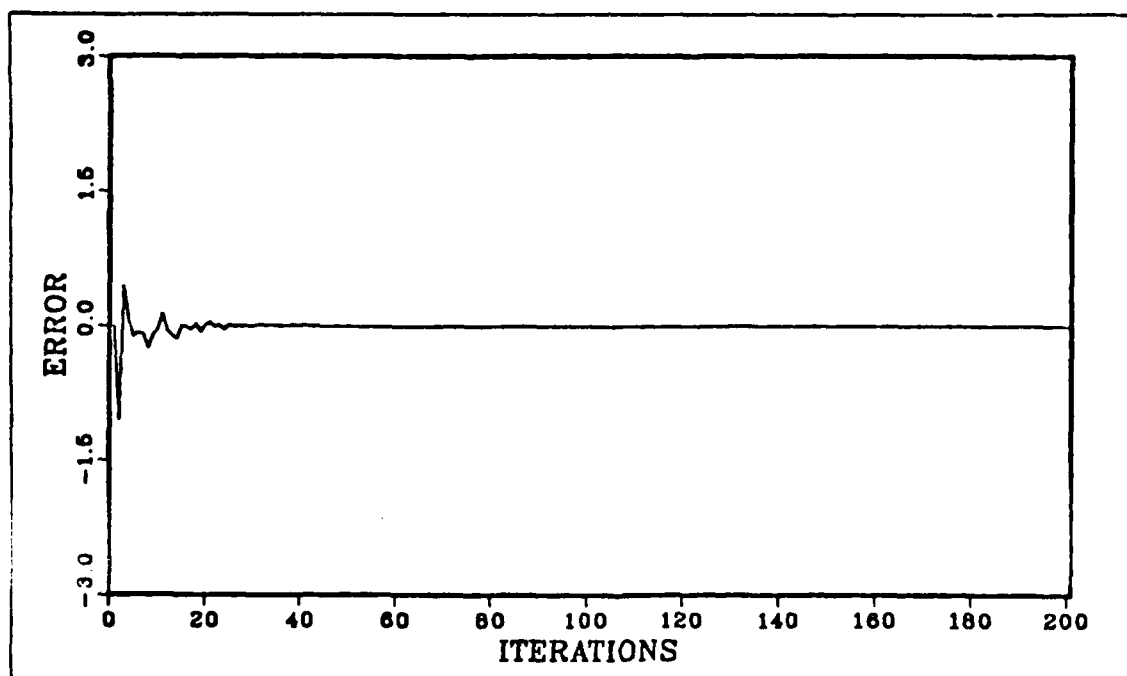


Figure 4.8 Learning Curve ($\mu = 0.5$).

V. ADAPTIVE LINEAR PHASE LATTICE ALGORITHM

A. INTRODUCTION

In Chapter III we dealt with the realization of fixed coefficient FIR lattice filter with both linear and nonlinear phase characteristics. We reviewed the basics of LMS algorithm and some adaptive lattice realizations reported in the literature in the previous chapter. The three adaptive lattice algorithms discussed are direct approximations of their non-adaptive counterparts. None of them estimates a gradient as required by the LMS algorithm. In this chapter we present a derivation for a new LMS based FIR adaptive lattice algorithm and extend it to the linear phase case as well. The new algorithm is then used in the estimation of spectral lines in white noise. Results of simulation are included.

B. LMS ALGORITHM FOR THE FIR LATTICE

From Eqs.(3.40) and (3.41) the FIR lattice equations can be written as

$$f_m(k) = s_m f_{m-1}(k) + k_m g_{m-1}(k-1) \quad (5.1)$$

$$g_m(k) = s_m g_{m-1}(k-1) + k_m f_{m-1}(k) \quad (5.2)$$

where $m=1,2,\dots,M$, $s_m=1$ for $m \neq M$ and k_m are the lattice reflection coefficients. A realization of Eqs.(5.1) and (5.2) is shown in Figure 5.1. The lattice input is $x(k)=g_0(k)=f_0(k)$. The output of the filter is

$$\begin{aligned} y(k) &= f_M(k) \\ &= s_M f_{M-1}(k) + k_M g_{M-1}(k-1) \end{aligned} \quad (5.3)$$

Therefore, the error, $e(k)$, is given by

$$\begin{aligned} e(k) &= d(k) - y(k) \\ &= d(k) - s_M f_{M-1}(k) - k_M g_{M-1}(k-1) \end{aligned} \quad (5.4)$$

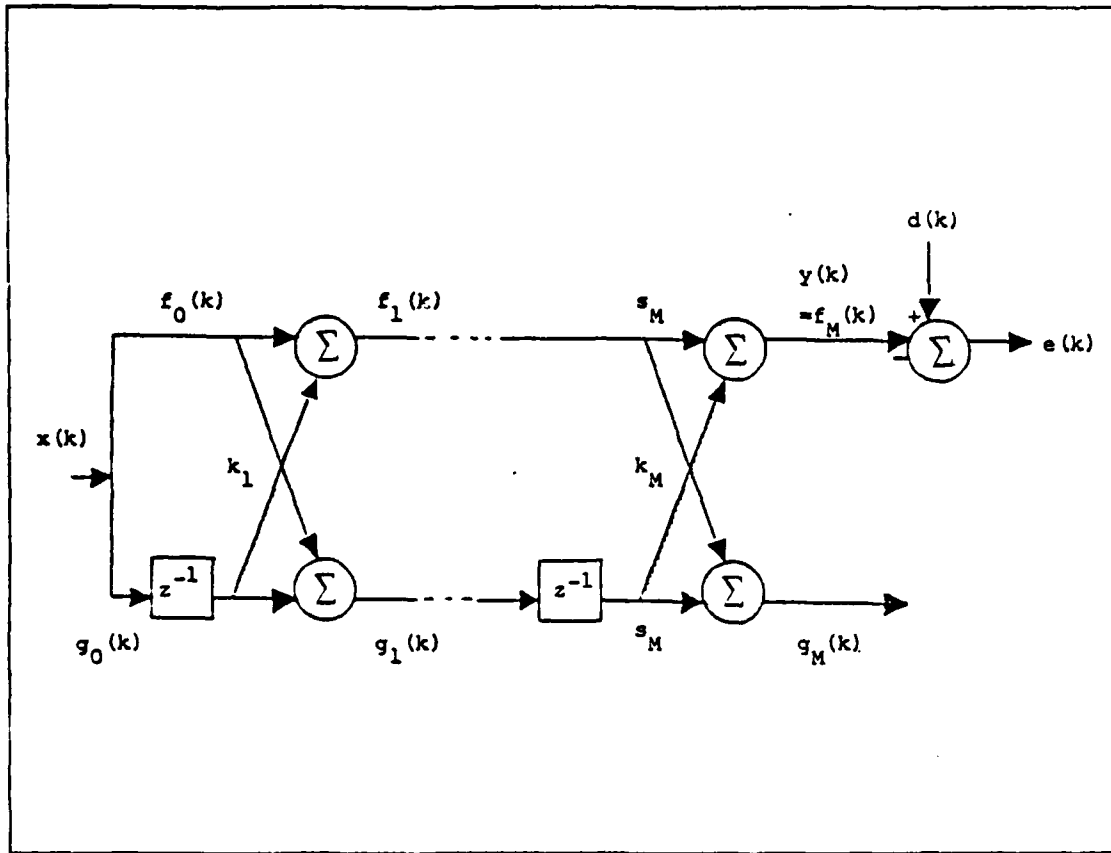


Figure 5.1 Adaptive FIR Lattice Filter.

where $d(k)$ is the desired signal. The objective is to minimize the mean square error

$$J = E \{ e^2(k) \} \quad (5.5)$$

The gradient of J with respect to k_m and s_m , respectively, are given by

$$\frac{\partial J}{\partial k_j} = -2 E \{ e(k) \frac{\partial y(k)}{\partial k_j} \} \quad (5.6)$$

and

$$\frac{\partial J}{\partial s_M} = -2 E\{e(k) f_{M-1}(k)\} \quad (5.7)$$

Then the LMS algorithm can be formulated as follows:

$$\begin{aligned} k_j(k+1) &= k_j(k) - \mu_k \left(\frac{\partial J}{\partial k_j} \right) \\ &= k_j(k) + 2 \mu_k E\{e(k) \left(\frac{\partial y(k)}{\partial k_j} \right)\} \\ &\cong k_j(k) + 2 \mu_k e(k) \left(\frac{\partial y(k)}{\partial k_j} \right) \end{aligned} \quad (5.8)$$

and

$$\begin{aligned} s_M(k+1) &= s_M(k) - \mu_s \left(\frac{\partial J}{\partial s_M} \right) \\ &= s_M(k) + 2 \mu_s E\{e(k) f_{M-1}(k)\} \\ &\cong s_M(k) + 2 \mu_s e(k) f_{M-1}(k) \end{aligned} \quad (5.9)$$

where μ_k and μ_s are the adaptation constant, and we have replaced the true gradient by its instantaneous estimate. Defining

$$z(k) = \frac{\partial y(k)}{\partial k_j} \quad (5.10)$$

yields

$$\begin{aligned} k_j(k+1) &= k_j(k) + 2 \mu_k e(k) z(k) \\ s_M(k+1) &= s_M(k) + 2 \mu_s e(k) f_{M-1}(k) \end{aligned} \quad (5.11)$$

where $j=1,2,\dots,M$. The next step is to estimate the gradient vector $z(k)$. For this purpose, define

$$\Phi_{i,j}(k) = \frac{\partial f_i(k)}{\partial k_j} \quad (5.12)$$

and

$$\Psi_{i,j}(k) = \frac{\partial g_i(k)}{\partial k_j} \quad (5.13)$$

Then from Eq.(5.3) $z(k)$ is given by

$$\begin{aligned} z(k) &= \Phi_{M,j}(k) \\ &= s_M(k) \Phi_{M-1,j}(k) + k_M(k) \Psi_{M-1,j}(k-1) + g_{M-1}(k-1) \delta_{M,j} \end{aligned} \quad (5.14)$$

where $\delta_{M,j} = (\partial k_M / \partial k_j)$. Substituting Eqs.(5.1) and (5.2) to Eqs.(5.12) and (5.13) then we have

$$\Phi_{i,j}(k) = s_i(k) \frac{\partial f_{i-1}(k)}{\partial k_j} + k_i(k) \frac{\partial g_{i-1}(k-1)}{\partial k_j} + g_{i-1}(k-1) \frac{\partial k_i}{\partial k_j}$$

$$\Psi_{i,j}(k) = s_i(k) \frac{\partial g_{i-1}(k-1)}{\partial k_j} + k_i(k) \frac{\partial f_{i-1}(k)}{\partial k_j} + f_{i-1}(k) \frac{\partial k_i}{\partial k_j}$$

therefore,

$$\Phi_{i,j}(k) = s_i(k) \Phi_{i-1,j}(k) + k_i(k) \Psi_{i-1,j}(k-1) + g_{i-1}(k-1) \delta_{i,j} \quad (5.15)$$

$$\Psi_{i,j}(k) = s_i(k) \Psi_{i-1,j}(k-1) + k_i(k) \Phi_{i-1,j}(k) + f_{i-1}(k) \delta_{i,j} \quad (5.16)$$

where

$$\delta_{i,j} = \frac{\partial k_i}{\partial k_j} \quad (5.17)$$

is the Kronecker delta function, and $i=1,2, \dots, M$ and $j=1,2, \dots, M$. If $i=0$, then Eqs.(5.15) and (5.16) are

$$\Phi_{0,j}(k) = \frac{\partial x(k)}{\partial k_j} = 0 \quad (5.18)$$

$$\Psi_{0,j}(k) = \Phi_{0,j}(k) = 0 \quad (5.19)$$

where $j=1,2, \dots, M$. Figure 5.2 shows the computation of gradient elements for the adaptive FIR lattice algorithm.

From the Eqs.(5.15), (5.16) and (5.19) we have

$$\Phi_{i,j}(k) = \Psi_{i,j}(k) = 0 \quad (5.20)$$

where $1 \leq i \leq j-1$. The computations of gradient elements at each case of $j=M, M-1, \dots, 1$ are as follows:

Case $j=M$:

$$\Phi_{0,M}(k) = \Psi_{0,M}(k) = 0$$

$$\Phi_{i,M}(k) = s_i(k)\Phi_{i-1,M}(k) - k_i(k)\Psi_{i-1,M}(k-1) + g_{i-1}(k-1)\delta_{i,M}$$

$$\Psi_{i,M}(k) = s_i(k)\Psi_{i-1,M}(k-1) + k_i(k)\Phi_{i-1,M}(k) + f_{i-1}(k)\delta_{i,M}$$

where $i=1,2, \dots, M$. From Eq.(5.20), every gradient element equals to zero except final stage elements and gradient elements of the final stage are

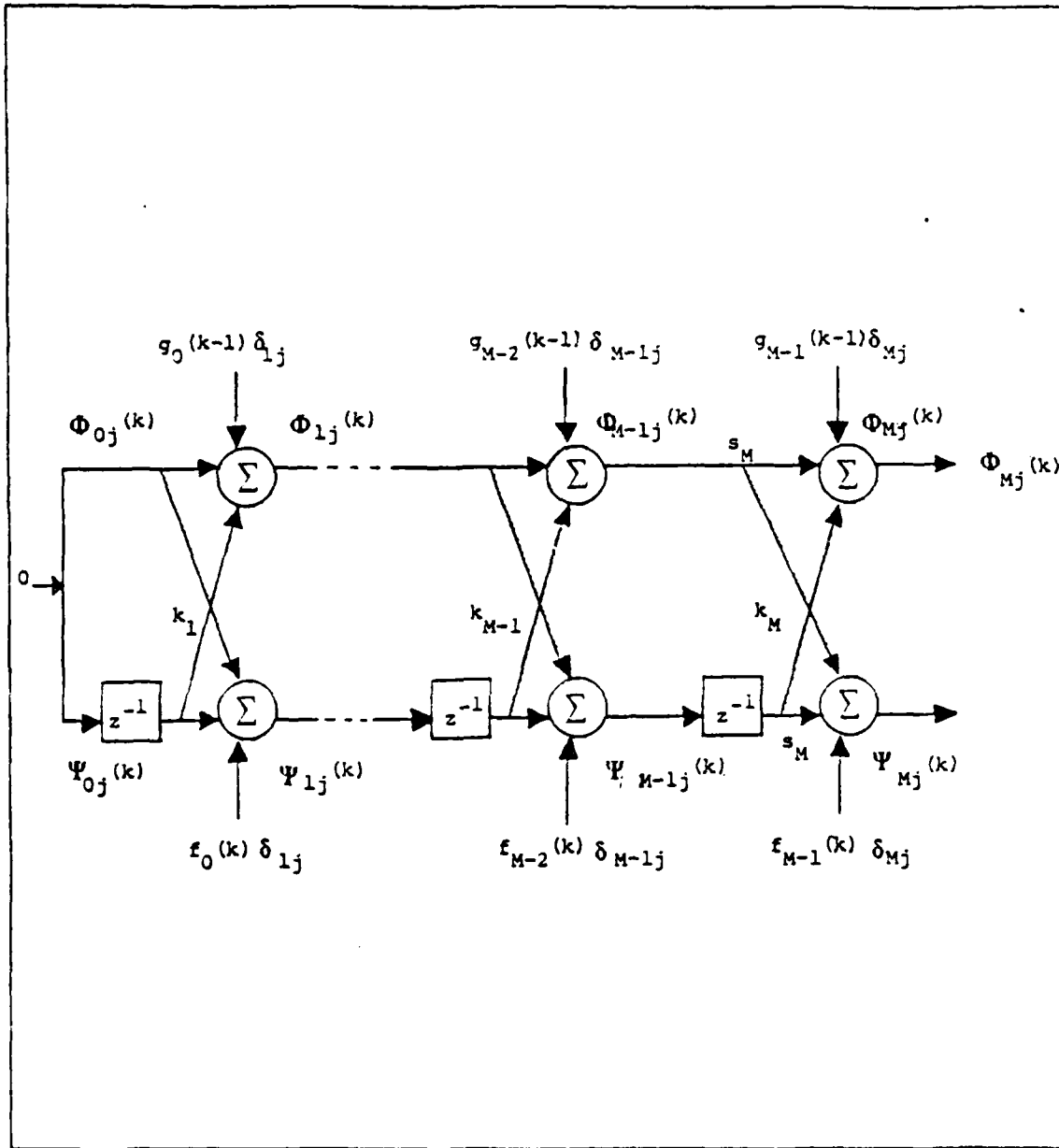


Figure 5.2 Computation of Gradient Elements for the Adaptive FIR Lattice Algorithm.

$$\Phi_{M,M}(k) = s_M(k)\Phi_{M-1,M}(k) + k_M(k)\Psi_{M-1,M}(k-1) + \varepsilon_{M-1}(k-1)\delta_{M,M}$$

$$\Psi_{M,M}(k) = s_M(k)\Psi_{M-1,M}(k-1) + k_M(k)\Phi_{M-1,M}(k) + \varepsilon_{M-1}(k)\delta_{M,M}$$

Applying the Eqs.(5.17) and (5.20) yields

$$\Phi_{M,M}(k) = g_{M-1}(k-1) \quad (5.21)$$

$$\Psi_{M,M}(k) = f_{M-1}(k) \quad (5.22)$$

Case $j = M-1$:

$$\Phi_{0,M-1}(k) = \Psi_{0,M-1}(k) = 0$$

$$\Phi_{i,M-1}(k) = s_i(k)\Phi_{i-1,M-1}(k) + k_i(k)\Psi_{i-1,M-1}(k-1) + g_{i-1}(k-1)\delta_{i,M-1}$$

$$\Psi_{i,M-1}(k) = s_i(k)\Psi_{i-1,M-1}(k-1) + k_i(k)\Phi_{i-1,M-1}(k) + f_{i-1}(k)\delta_{i,M-1}$$

where $i = 1, 2, \dots, M$. Applying the Eq.(5.20), last two stage elements are considerable, if $i = M-1$, then we have

$$\Phi_{M-1,M-1}(k) = s_{M-1}(k)\Phi_{M-2,M-1}(k) + k_{M-1}(k)\Psi_{M-2,M-1}(k-1) + g_{M-2}(k-1)\delta_{M-1,M-1}$$

$$\Psi_{M-1,M-1}(k) = s_{M-1}(k)\Psi_{M-2,M-1}(k-1) + k_{M-1}(k)\Phi_{M-2,M-1}(k) + f_{M-2}(k)\delta_{M-1,M-1}$$

Applying the Eqs.(5.17) and (5.20) yields

$$\Phi_{M-1,M-1}(k) = g_{M-2}(k-1) \quad (5.23)$$

$$\Psi_{M-1,M-1}(k) = f_{M-2}(k) \quad (5.24)$$

And the last stage terms are

$$\Phi_{M,M-1}(k) = s_M(k)\Phi_{M-1,M-1}(k) + k_M(k)\Psi_{M-1,M-1}(k-1) + g_{M-1}(k-1)\delta_{M,M-1}$$

$$\Psi_{M,M-1}(k) = s_M(k)\Psi_{M-1,M-1}(k-1) + k_M(k)\Phi_{M-1,M-1}(k) + f_{M-1}(k)\delta_{M,M-1}$$

Using the Eqs.(5.17), (5.23) and (5.24) yields

$$\Phi_{M,M-1}(k) = s_M(k)g_{M-2}(k-1) + k_M(k) f_{M-2}(k-1) \quad (5.25)$$

$$\Psi_{M,M-1}(k) = s_M(k)f_{M-2}(k-1) + k_M(k) g_{M-2}(k-1) \quad (5.26)$$

Finally,

Case $j=1$:

$$\Phi_{0,1}(k) = \Psi_{0,1}(k) = 0$$

$$\Phi_{i,1}(k) = s_i(k)\Phi_{i-1,1}(k) + k_i(k)\Psi_{i-1,1}(k-1) + g_{i-1}(k-1)\delta_{i,1} \quad (5.27)$$

$$\Psi_{i,1}(k) = s_i(k)\Psi_{i-1,1}(k-1) + k_i(k)\Phi_{i-1,1}(k) + f_{i-1}(k)\delta_{i,1} \quad (5.28)$$

where $i=1,2, \dots, M$.

The LMS algorithm derived in the foregoing can be summarized in Table 1.

Simulation Results :

The performance of the algorithm summarized in Table 1 has been observed by computer simulation. The configuration used is the system identification experiment as discussed in Section (IV. D) using the same fixed coefficient filter. The learning curves obtained using the new algorithm are shown in Figures 5.3 to 5.5. Comparing these with the learning curves in Figures 4.6 to 4.8, which are obtained using approximate algorithms, we observe significantly faster convergence rate for the new algorithm.

TABLE 1
LMS ALGORITHM FOR THE FIR LATTICE

Initialization:

$$\underline{k}(0) = 0$$

$$s_M(0) = 1$$

Lattice:

$$x(k) = f_0(k) = g_0(k)$$

$$f_m(k) = s_m(k)f_{m-1}(k) + k_m(k)g_{m-1}(k-1)$$

$$g_m(k) = s_m(k)g_{m-1}(k-1) + k_m(k)f_{m-1}(k)$$

$$m = 1, 2, \dots, M$$

$$y(k) = f_M(k)$$

Update Equations:

$$k_j(k+1) = k_j(k) + 2 [\mu_k / \sigma_{k_j}^2(k)] e(k) z_j(k)$$

$$s_M(k+1) = s_M(k) + 2 [\mu_s / \sigma_s^2(k)] e(k) f_{M-1}(k)$$

where μ_k and μ_s are the adaptation constants,

$$\sigma_{k_j}^2(k) = \lambda \sigma_{k_j}^2(k-1) + (1-\lambda) \Phi_{M,j}^2(k)$$

and

$$\sigma_s^2(k) = \lambda \sigma_s^2(k-1) + (1-\lambda) f_{M-1}^2(k)$$

are estimations of power in $z_j(k)$ and $f_{M-1}(k)$, respectively and λ is a positive weighting constant, $0 < \lambda \leq 1$.

Gradient Vector Elements:

$$\Phi_{0,j}(k) = \Psi_{0,j}(k) = 0$$

$$\begin{aligned} \Phi_{i,j}(k) = & s_i(k)\Phi_{i-1,j}(k) + k_i(k)\Psi_{i-1,j}(k-1) \\ & + g_{i-1}(k-1)\delta_{i,j} \end{aligned}$$

$$\begin{aligned} \Psi_{i,j}(k) = & s_i(k)\Psi_{i-1,j}(k-1) + k_i(k)\Phi_{i-1,j}(k) \\ & + f_{i-1}(k)\delta_{i,j} \end{aligned}$$

$$z_j(k) = \Phi_{M,j}(k)$$

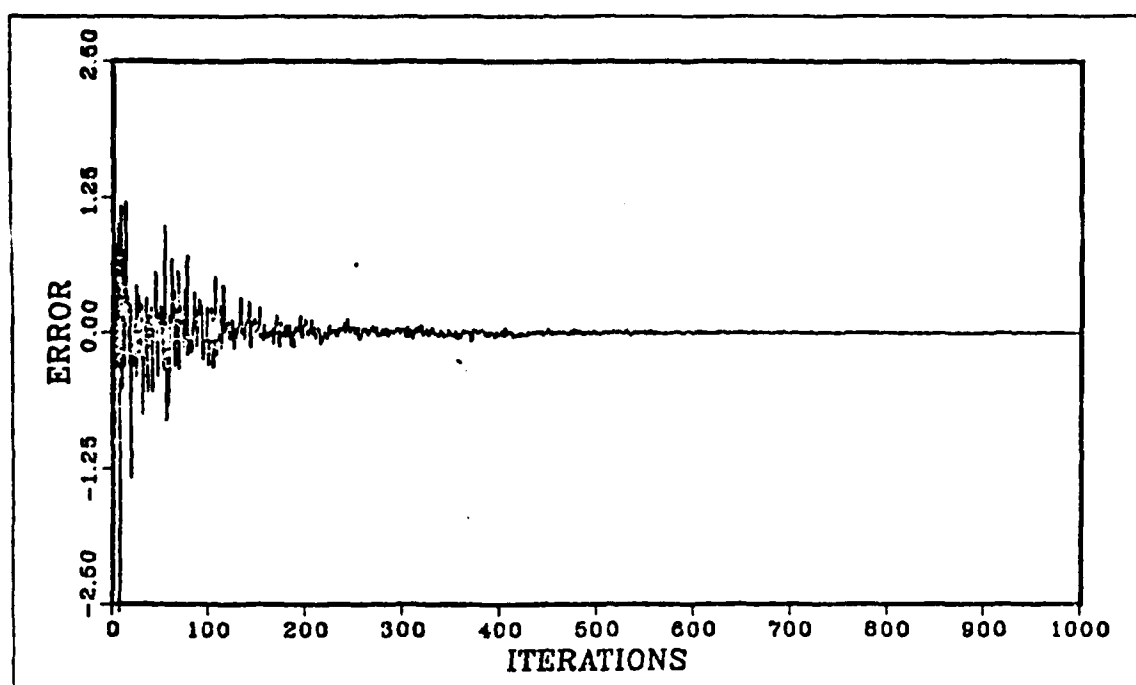


Figure 5.3 Learning Curve ($\mu = 0.1$).

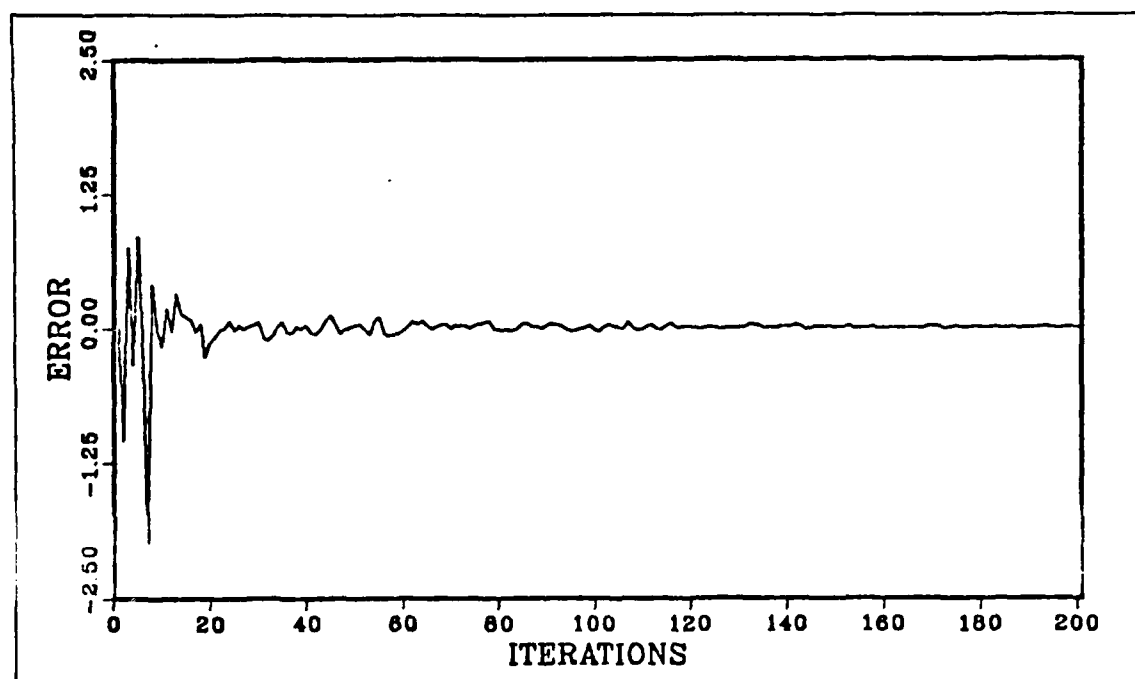


Figure 5.4 Learning Curve ($\mu = 0.3$).

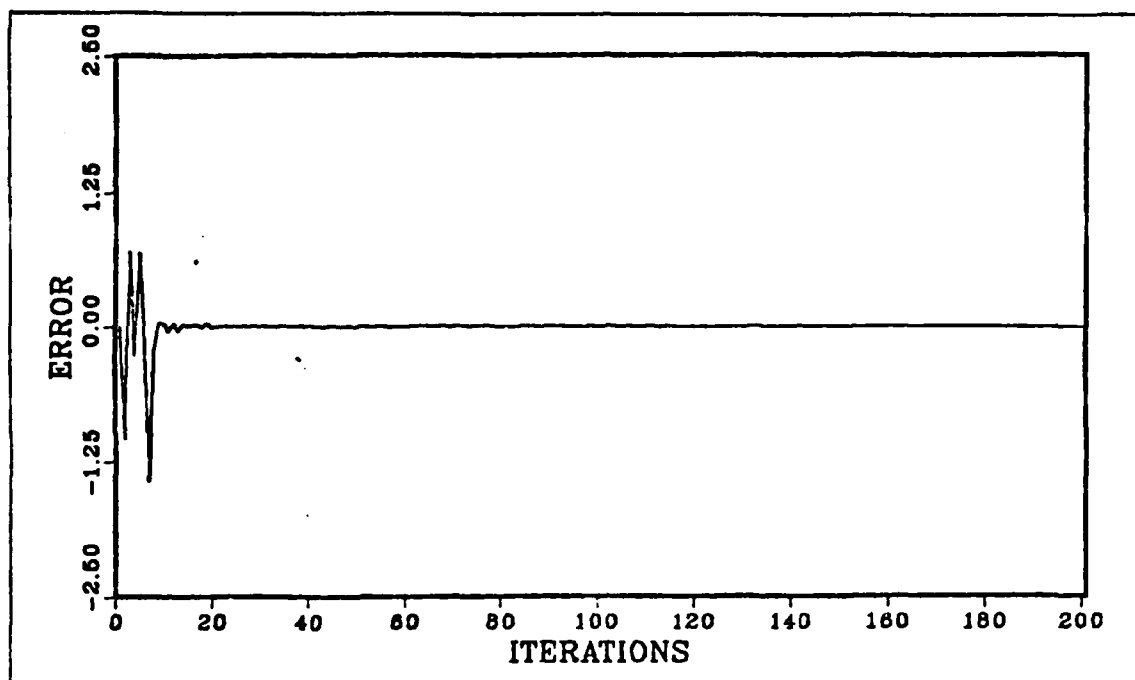


Figure 5.5 Learning Curve ($\mu = 0.5$).

C. LINEAR PHASE FIR LATTICE ALGORITHM

We now extend the LMS algorithm derived in the previous section to the linear phase FIR lattice filter.

From Eq.(3.29), output of the linear phase FIR lattice can be written as

$$y(k) = f_M(k) + g_M(k-D) \quad (5.29)$$

The lattice input is $x(k) = f_0(k) = g_0(k)$. Substituting Eqs.(5.1) and (5.2) in the output equation of the filter yields

$$\begin{aligned} y(k) &= s_M f_{M-1}(k) + k_M g_{M-1}(k-1) + s_M g_{M-1}(k-D-1) + k_M f_{M-1}(k-D) \\ &= s_M [f_{M-1}(k) + g_{M-1}(k-D-1)] + k_M [f_{M-1}(k-D) + g_{M-1}(k-1)] \end{aligned} \quad (5.30)$$

and the error, $e(k)$, is given by

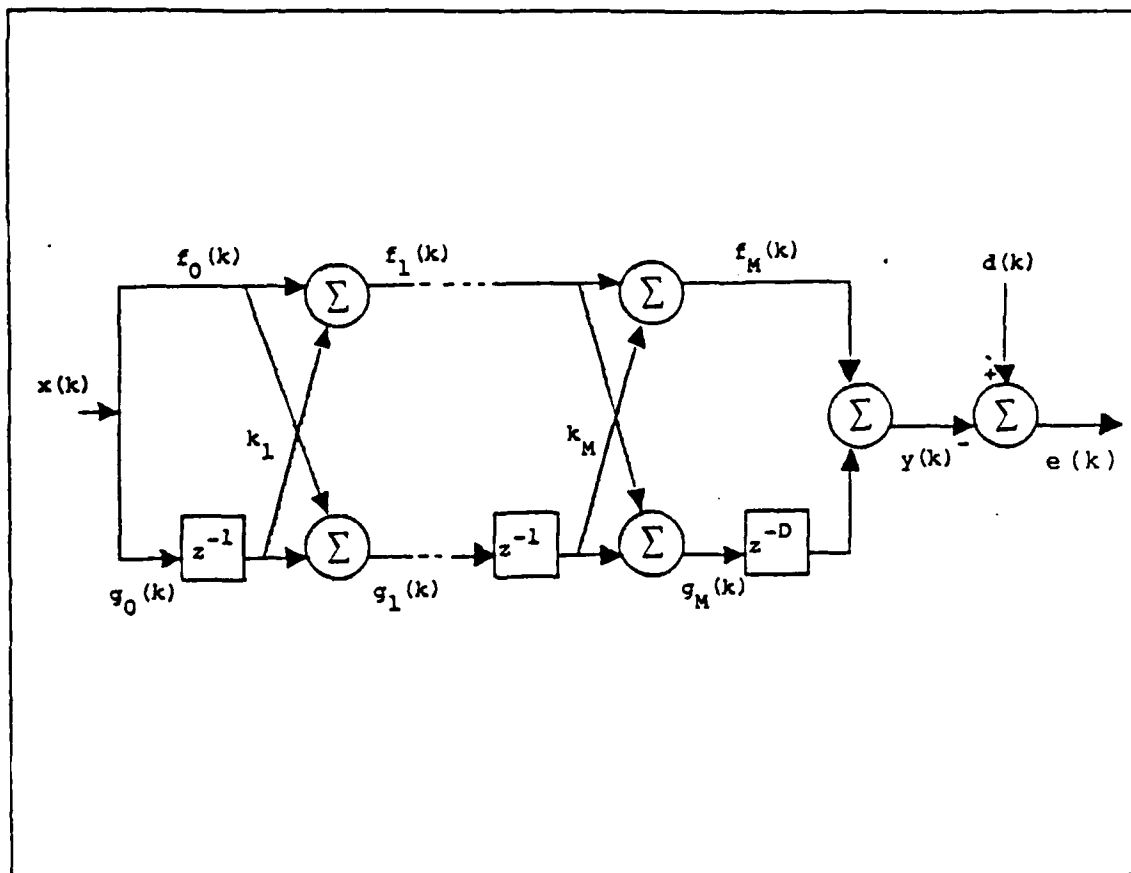


Figure 5.6 Adaptive Linear Phase FIR Lattice Filter.

$$\begin{aligned}
 e(k) &= d(k) - y(k) \\
 &= d(k) - s_M [f_{M-1}(k) + g_{M-1}(k-D-1)] - k_M [f_{M-1}(k-D) + g_{M-1}(k-1)] \quad (5.31)
 \end{aligned}$$

where $d(k)$ is the desired signal. A schematic of the linear phase FIR lattice realization is shown in Figure 5.6. From Eq.(5.31) we see that $y(k)$ is a function of both s_M and k_M . But $f_{M-1}(k)$ and $g_{M-1}(k-1)$ are functions of k_{M-1} and so on. Therefore, in order to minimize the mean square error, we define a cost function

$$J = E \{ e^2(k) \} \quad (5.32)$$

and a gradient element

$$z(k) = \frac{\partial y(k)}{\partial k_j} \quad (5.33)$$

Then following the treatment in Eqs.(5.6)-(5.11) the LMS algorithm for the linear phase lattice can be written as

$$\begin{aligned} k_j(k+1) &= k_j(k) + 2 \mu_k e(k) z(k) \\ s_M(k+1) &= s_M(k) + 2 \mu_s e(k) [f_{M-1}(k) + g_{M-1}(k-D-1)] \end{aligned} \quad (5.34)$$

where $j=1,2,\dots,M$. From Eqs.(5.29) and (5.30), the gradient vector $z(k)$ is given by

$$\begin{aligned} z(k) &= \Phi_{M,j}(k) + \Psi_{M,j}(k-D) \\ &= s_M(k) [\Phi_{M-1,j}(k) + \Psi_{M-1,j}(k-D-1)] + k_M(k) [\Phi_{M-1,j}(k-D) \\ &\quad + \Psi_{M-1,j}(k-1)] + [f_{M-1}(k-D) + g_{M-1}(k-1)] \delta_{M,j} \end{aligned} \quad (5.35)$$

where $\delta_{M,j} = (\partial k_M / \partial k_j)$, and $\Phi_{m,j}$ and $\Psi_{m,j}$ are obtained on the same lines as in Eqs.(5.15)-(5.17). The resulting LMS algorithm is summarized in Table 2.

Simulation Results :

The performance of the algorithm summarized in Table 2 has been observed by computer simulation. The configuration used is the system identification experiment as discussed in Section (IV. D). We have used two different fixed coefficient linear phase FIR filter examples, one with N even and the other with N odd. They are

$$H_3(z) = 0.154 + 0.462z^{-1} + 0.462z^{-2} + 0.154z^{-3}$$

and

$$H_4(z) = 0.15 - 0.45z^{-1} + 0.36z^{-2} - 0.45z^{-3} + 0.15z^{-4}$$

The learning curves obtained using the new algorithm are shown in Figures 5.7 to 5.10. Figures 5.7 and 5.8 are obtained learning curves with linear phase FIR transfer function $H_3(z)$. Figures 5.9 and 5.10 are obtained learning curves with linear phase FIR transfer function $H_4(z)$.

TABLE 2
LINEAR PHASE FIR LATTICE ALGORITHM

Initialization:

$$K(0) = 0$$

$$s_M(0) = 1$$

Lattice:

$$x(k) = f_0(k) = g_0(k)$$

$$f_m(k) = s_m(k)f_{m-1}(k) + k_m(k)g_{m-1}(k-1)$$

$$g_m(k) = s_m(k)g_{m-1}(k-1) + k_m(k)f_{m-1}(k)$$

$$m = 1, 2, \dots, M$$

$$y(k) = s_M [f_{M-1}(k) + g_{M-1}(k-D-1)] + k_M [f_{M-1}(k-D) + g_{M-1}(k-1)]$$

Update Equations:

$$k_j(k+1) = k_j(k) + 2 [\mu_k / \sigma_{k_j}^2(k)] e(k) z_j(k)$$

$$s_M(k+1) = s_M(k) + 2 [\mu_s / \sigma_s^2(k)] e(k) [f_{M-1}(k) + g_{M-1}(k-D-1)]$$

where μ_k and μ_s are the adaptation constants,

$$\sigma_{k_j}^2(k) = \lambda \sigma_{k_j}^2(k-1) + (1-\lambda) [\Phi_{M,j}(k) + \Psi_{M,j}(k-D)]^2$$

and

$$\sigma_s^2(k) = \lambda \sigma_s^2(k-1) + (1-\lambda) [f_{M-1}(k) + g_{M-1}(k-D-1)]^2$$

are estimations of power in $z_j(k)$ and $[f_{M-1}(k) + g_{M-1}(k-D-1)]$, respectively and λ is a positive weighting constant, $0 < \lambda \leq 1$.

Gradient Vector Elements:

$$\Phi_{0,j}(k) = \Psi_{0,j}(k) = 0$$

$$\Phi_{i,j}(k) = s_i(k)\Phi_{i-1,j}(k) + k_i(k)\Psi_{i-1,j}(k-1) + g_{i-1}(k-1)\delta_{i,j}$$

$$\Psi_{i,j}(k) = s_i(k)\Psi_{i-1,j}(k-1) + k_i(k)\Phi_{i-1,j}(k) + f_{i-1}(k)\delta_{i,j}$$

$$z_j(k) = \Phi_{M,j}(k) + \Psi_{M,j}(k-D)$$

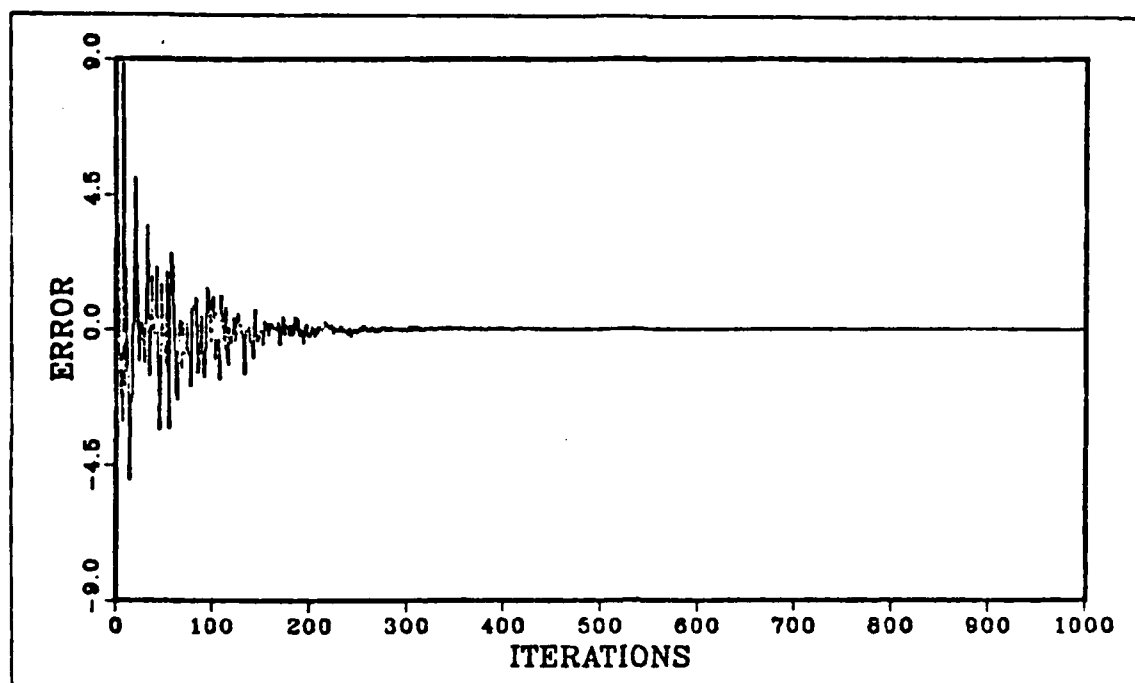


Figure 5.7 Learning Curve ($\mu = 0.1$).

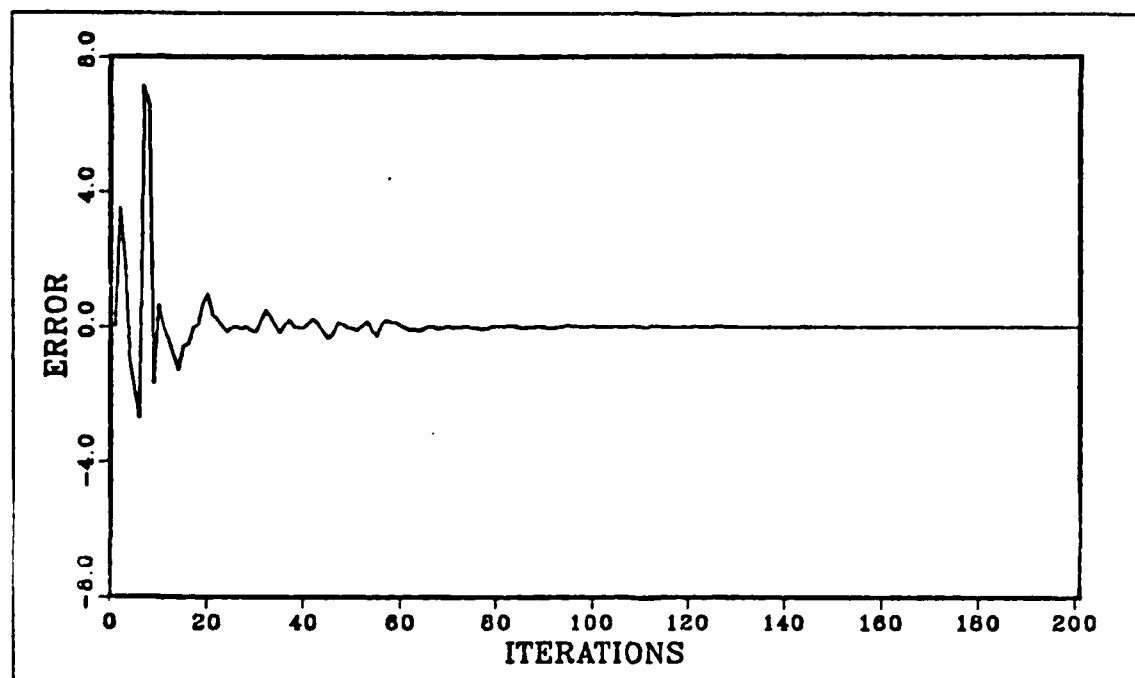


Figure 5.8 Learning Curve ($\mu = 0.3$).

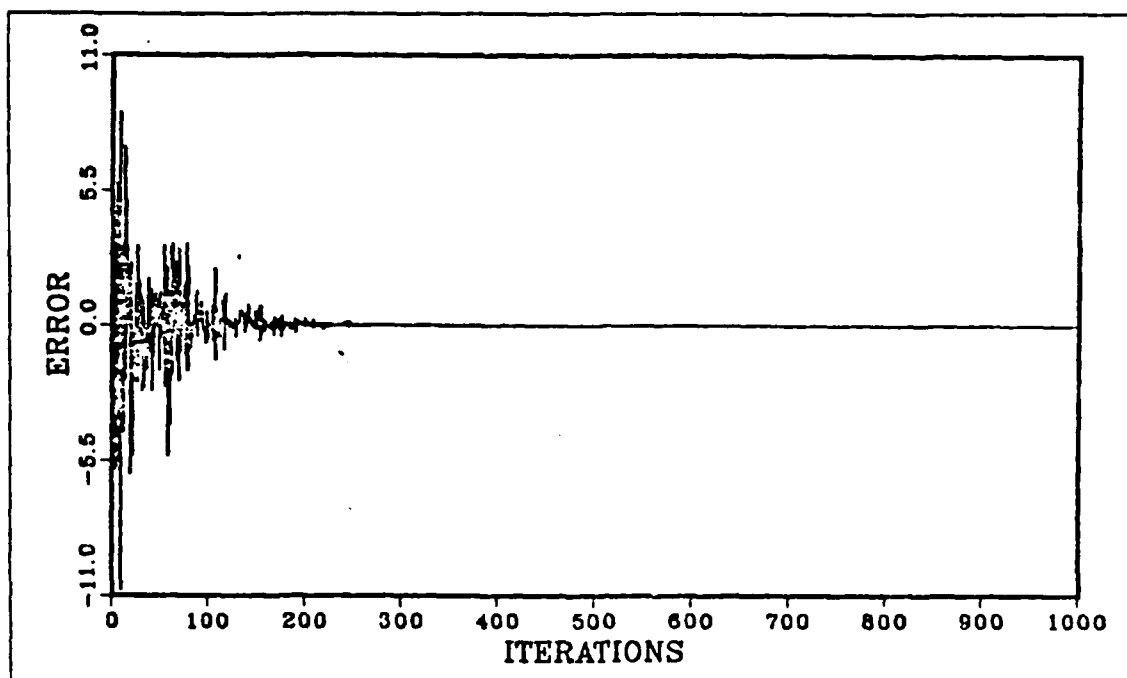


Figure 5.9 Learning Curve ($\mu = 0.03$).

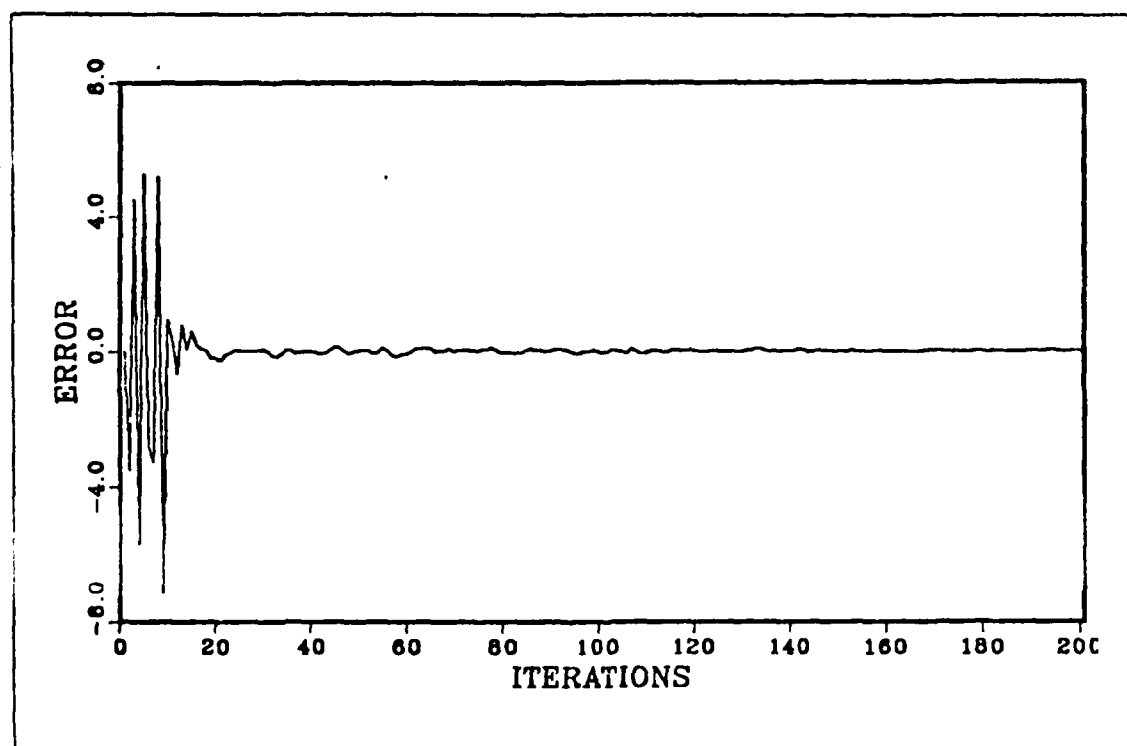


Figure 5.10 Learning Curve ($\mu = 0.1$).

D. SPECTRAL ESTIMATION

In this section, we extend the linear phase FIR adaptive lattice algorithm derived in the previous section to the estimation of spectral lines in white noise. The spectral estimation problem is somewhat different from the system identification experiment that we have been considering so far. In a typical spectral estimation problem, we are given a time series and we need to estimate its spectrum. A suitable configuration that is frequently used for this purpose is shown in Figure 5.11. Comparing this with Figure 4.5, we notice that we have the given time series at the input of the adaptive filter and the filter updates its coefficients to minimize the mean square output. In doing this, we assume that the input process $x(k)$ has been generated by passing a white noise sequence through a filter, say $I(z)$, and the adaptive filter attempts to realize an inverse filter, say $H(z) = 1/I(z)$. Considering that the adaptive filter has sufficient degrees of freedom, the output of the filter will be a white noise sequence.

The adaptive algorithm summarized in Table 2 can be used in spectral estimation after appropriate modification. In the present case we minimize the cost function,

$$J = E\{y^2(k)\}$$

rather than $J = E\{e^2(k)\}$. The resulting update equations can be shown to be

$$k_j(k+1) = k_j(k) - 2 [\mu_k / \sigma_{k_j}^2(k)] e(k) z_j(k) \quad (5.36)$$

$$s_M(k+1) = s_M(k) - 2 [\mu_s / \sigma_s^2(k)] e(k) [f_{M-1}(k) + g_{M-1}(k-D-1)] \quad (5.37)$$

where $z_j(k)$, $\sigma_{k_j}(k)$ and $\sigma_s(k)$ are as defined in Table 2.

Using the adaptive algorithm in Eqs.(5.36) and (5.37) we can obtain values of the reflection coefficients and the s_M coefficient. We then obtain the equivalent polynomial, $B(z)$, by going through the standard step up procedure given by [Ref. 1]:

$$b_{m,0} = s_m$$

$$b_{m,m} = k_m$$

$$b_{m,i} = s_m b_{m-1,i} + k_m b_{m-1,m-i}$$

where $i=1,2,\dots,m-1$ and $m=1,2,\dots,M$. Finally the required linear phase transfer function is obtained as follows:

$$H_{N-1}(z) = F_M(z) + z^{-D} G_M(z)$$

where $F_M(z) = B(z)$ and $G_M(z) = z^{-M} B(z^{-1})$. The spectrum is computed as follows:

$$S(f) = \frac{1}{|h_0 + h_1 e^{-j\omega} + \dots + h_{N-1} e^{-j(n-1)\omega}|^2}$$

where $\omega = 2\pi f$, and f is the normalized frequency (with respect to the sampling frequency) in the range, $0 \leq f < 0.5$.

Simulation Results :

The input process $x(k)$ consists of a signal in noise, given by

$$x(k) = s(k) + w(k)$$

where $s(k)$ may be a single, or multiple sinusoids and $w(k)$ is a zero mean unit variance white noise sequence. In the following we consider several examples using parameters ranging from a single sinusoid to 4 sinusoids, SNRs from 30dB to 10dB, and filter order, M , from 2 to 30.

Example 1: We consider a single sinusoid given by

$$x(k) = \sqrt{2} \cos(2\pi f k) + w(k) \quad (5.38)$$

where $f = 0.15$, $\text{SNR} = 30\text{dB}$.

Figures 5.12, 5.13, 5.14, and 5.15 are plots of frequency spectrum with different order of lattice M and adaptation constant μ .

Figure 5.12 is the plot of the case $M = 2$ and $\mu = 0.015$. We see that the peak is at $f = 0.15$ and no spurious responses.

Figure 5.13 shows the frequency spectrum of $M = 4$ and $\mu = 0.01$. We observe that one peak is at $f = 0.15$ and a spurious response whose magnitude is about -53dB at $f = 0.37$.

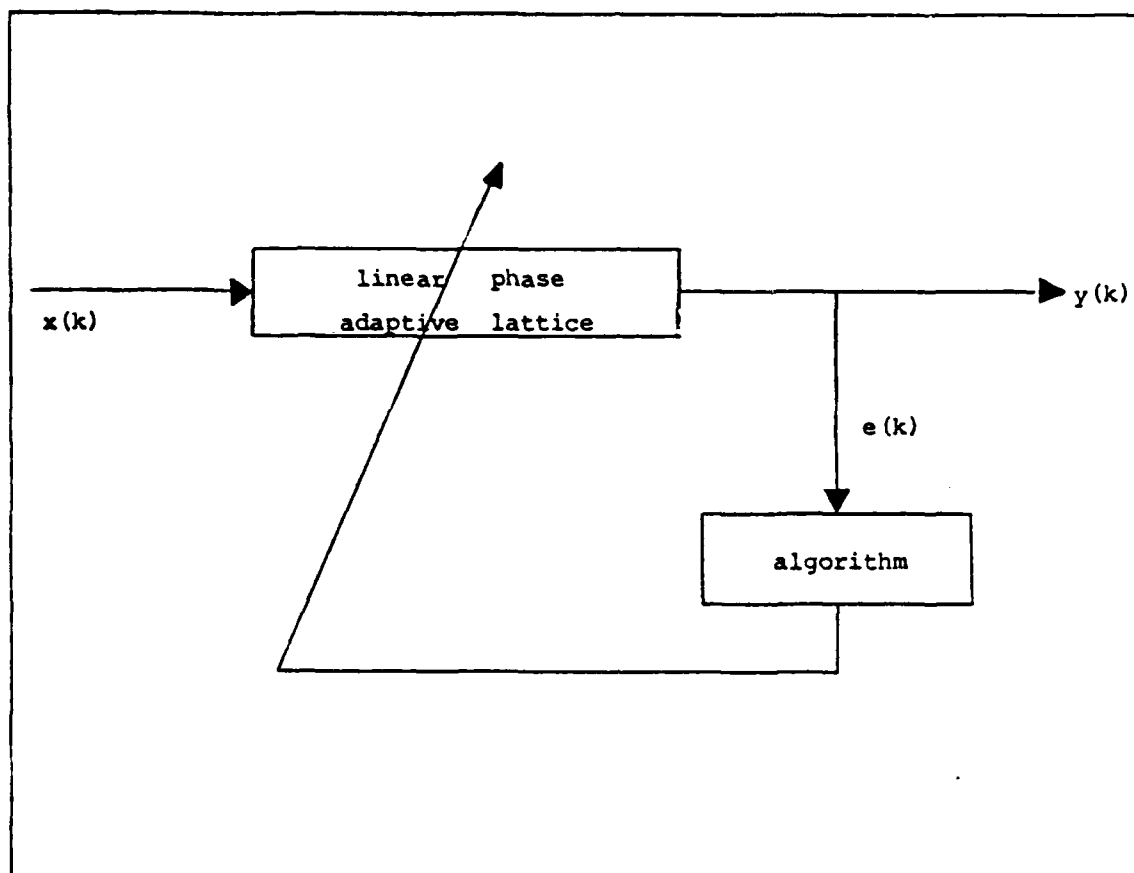


Figure 5.11 Spectral Estimation Mode.

Figure 5.14 shows the frequency spectrum of $M = 10$ and $\mu = 0.03$. There are one peak at $f = 0.15$ and four spurious responses at $f = 0.05, 0.25, 0.35$ and 0.45 . The largest spurious response is about -15 dB at $f = 0.45$.

Figure 5.15 shows the frequency spectrum of $M = 20$ and $\mu = 0.03$. There are one peak at $f = 0.15$ and nine spurious responses. The largest spurious response is about -27 dB at $f = 0.125$.

Through the Figures 5.12, 5.13, 5.14, and 5.15 we can determine that the best model order to detect the one sinusoidal signal is $M = 2$.

Next, we fix the order of lattice M , adaptation constant μ and iteration number k . Figures 5.16, 5.17 and 5.18 are the plots of frequency spectrum with changing the signal to noise ratio (SNR).

Figure 5.16 is the plot of $M = 20$, $\mu = 0.05$, $k = 3000$ and $\text{SNR} = 30\text{dB}$. There is a peak at $f = 0.15$ and the largest spurious response is about -27dB at $f = 0.125$.

Figure 5.17 is the plot of $M = 20$, $\mu = 0.05$, $k = 5000$ and $\text{SNR} = 20\text{dB}$. The largest spurious response at $f = 0.425$ is larger than the desired response. At this case, we cannot estimate the frequency of sinusoid.

Figure 5.18 is the plot of $M = 20$, $\mu = 0.05$, $k = 3000$ and $\text{SNR} = 10\text{dB}$. Three of the spurious responses are larger than the desired response. The desired response is about -13dB .

From Figures 5.16, 5.17 and 5.18, if the SNR is getting smaller then estimating sinusoidal frequency is more difficult.

Example 2: Next, to estimate two sinusoidal frequencies in noise, set the input $x(k)$ as

$$x(k) = \sqrt{2} \{ \cos(2\pi f_1 k) + \cos(2\pi f_2 k) \} + w(k) \quad (5.39)$$

where the normalized frequencies of signals $f_1 = 0.15$ and $f_2 = 0.25$ and set $\text{SNR} = 30\text{dB}$.

Figure 5.19 shows the frequency spectrum of $M = 4$ and $\mu = 0.02$. There are two peaks at $f_1 = 0.15$ and $f_2 = 0.25$ and no spurious responses.

Figure 5.20 shows the frequency spectrum of $M = 20$ and $\mu = 0.022$. There are two peaks at $f_1 = 0.15$ and $f_2 = 0.25$, and eight spurious responses. The largest spurious response is about -20dB at $f = 0.375$.

From Figures 5.19 and 5.20, the best model order to estimate the frequencies of two sinusoidal signals is $M = 4$.

Example 3: Next, to estimate four sinusoidal frequencies in noise, set the input $x(k)$ as

$$x(k) = \sqrt{2} \{ \cos(2\pi f_1 k) + \cos(2\pi f_2 k) + \cos(2\pi f_3 k) + \cos(2\pi f_4 k) \} + w(k) \quad (5.40)$$

where the normalized frequencies of signals $f_1 = 0.05$, $f_2 = 0.15$, $f_3 = 0.25$, $f_4 = 0.35$, and set $\text{SNR} = 30\text{dB}$.

Figure 5.21 shows the frequency spectrum of $M = 8$ and $\mu = 0.015$. There are four peaks at $f_1 = 0.07$, $f_2 = 0.15$, $f_3 = 0.27$ and $f_4 = 0.375$, and no spurious responses.

Figure 5.22 shows the frequency spectrum of $M = 30$ and $\mu = 0.014$. There are four peaks at $f_1 = 0.05$, $f_2 = 0.15$, $f_3 = 0.25$ and $f_4 = 0.35$, and eleven spurious responses. The largest spurious response is about -23dB at $f = 0.45$.

From Figures 5.21 and 5.22, the best model order to estimate the frequencies of four sinusoidal signals is $M = 8$.

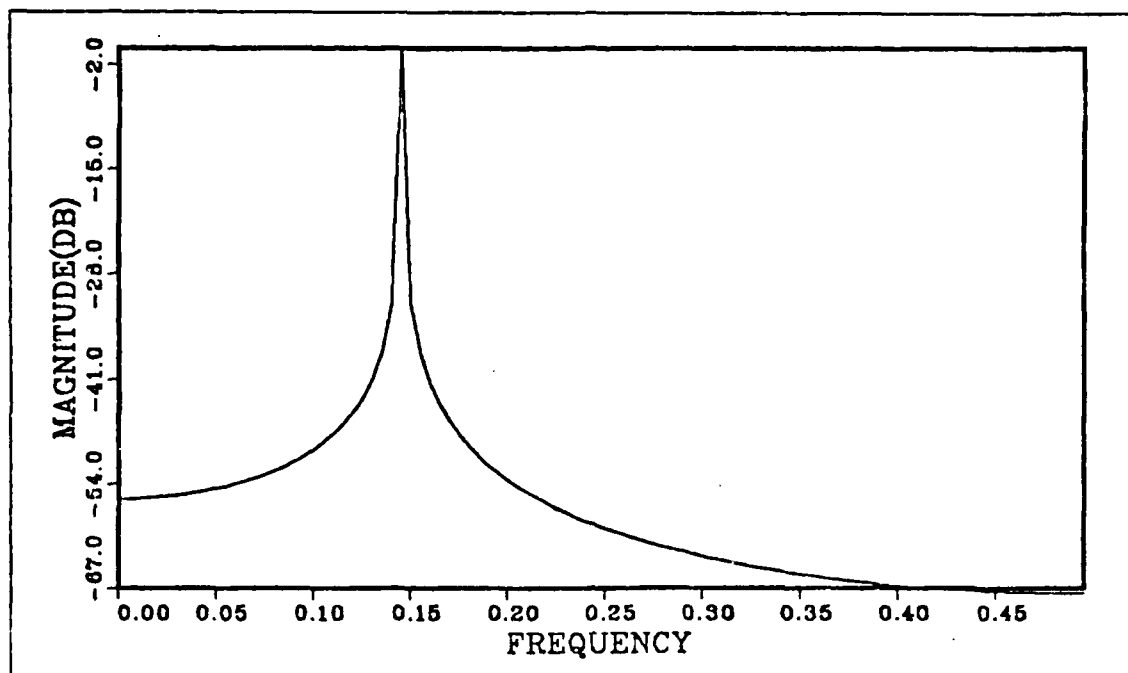


Figure 5.12 Frequency Spectrum (1 sinusoid, $M = 2$, $\mu = 0.015$).

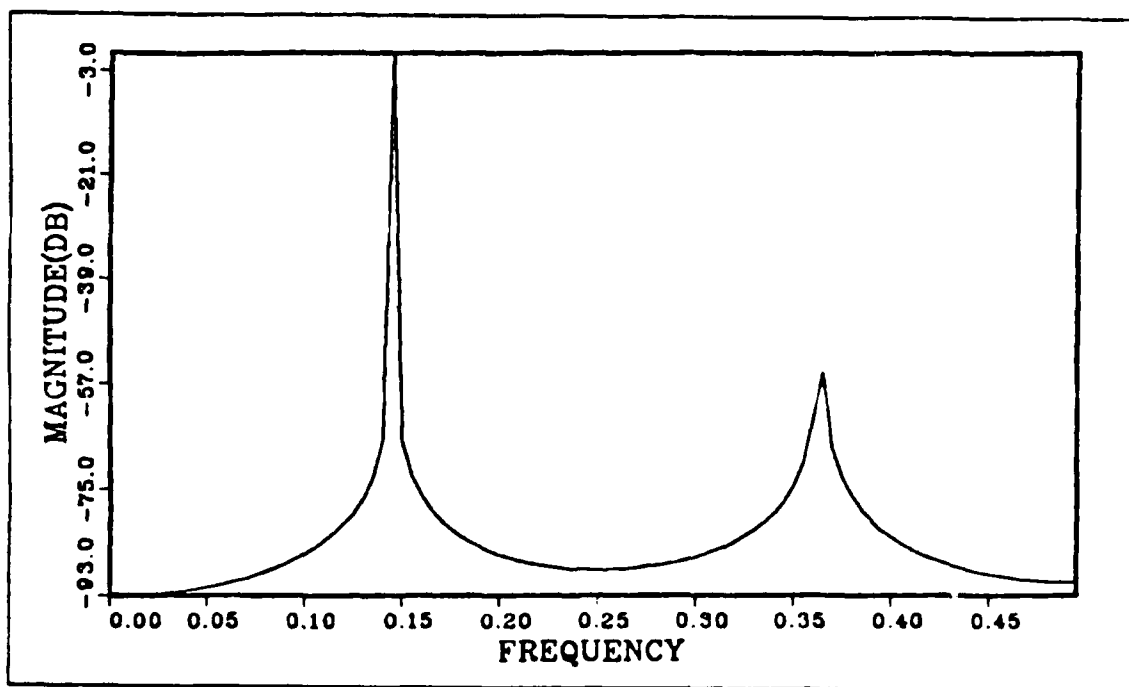


Figure 5.13 Frequency Spectrum (1 sinusoid, $M = 4$, $\mu = 0.01$).

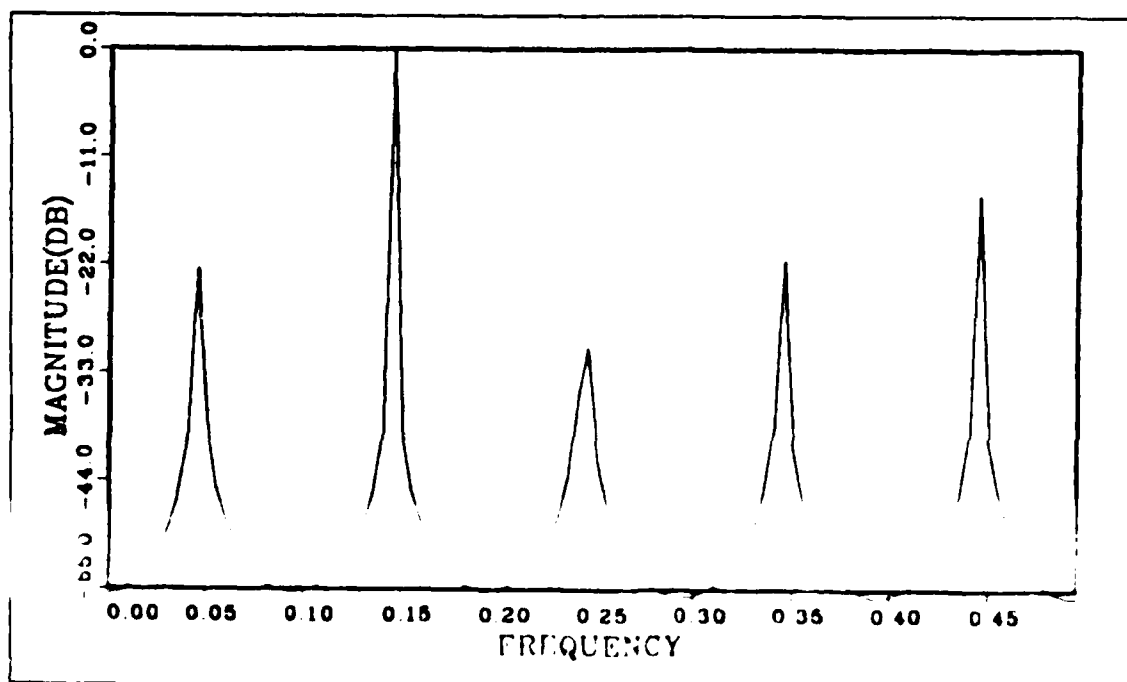


Figure 5.14 Frequency Spectrum (1 sinusoid, $M = 10$, $\mu = 0.03$).

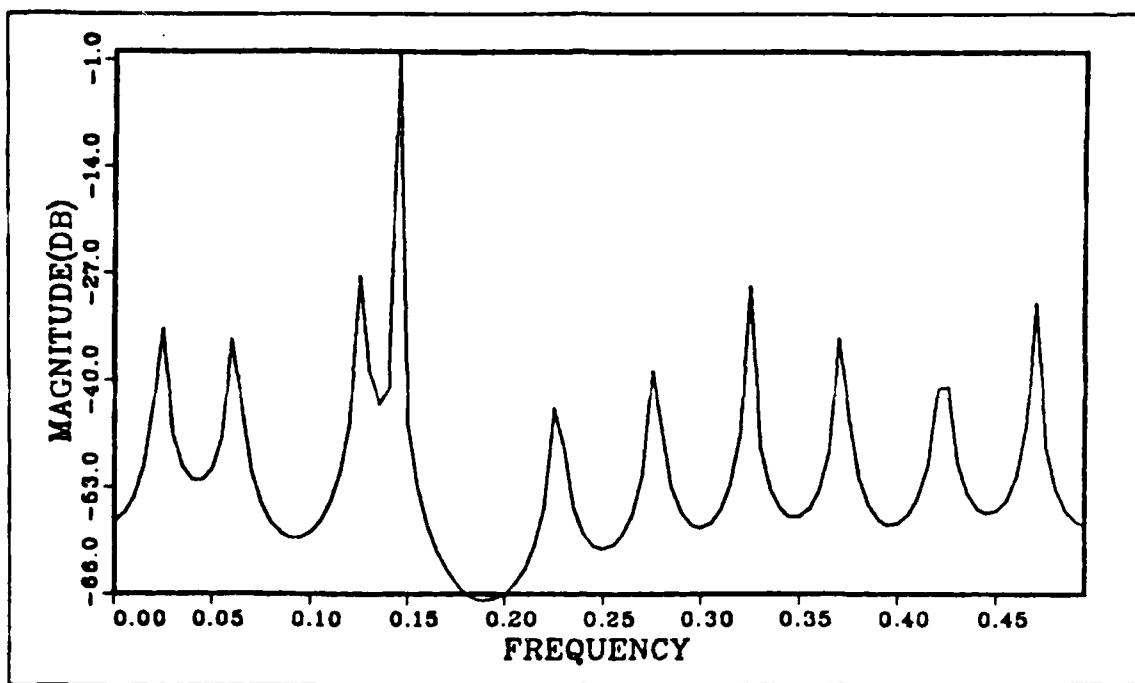


Figure 5.15 Frequency Spectrum (1 sinusoid, $M = 20$, $\mu = 0.03$).

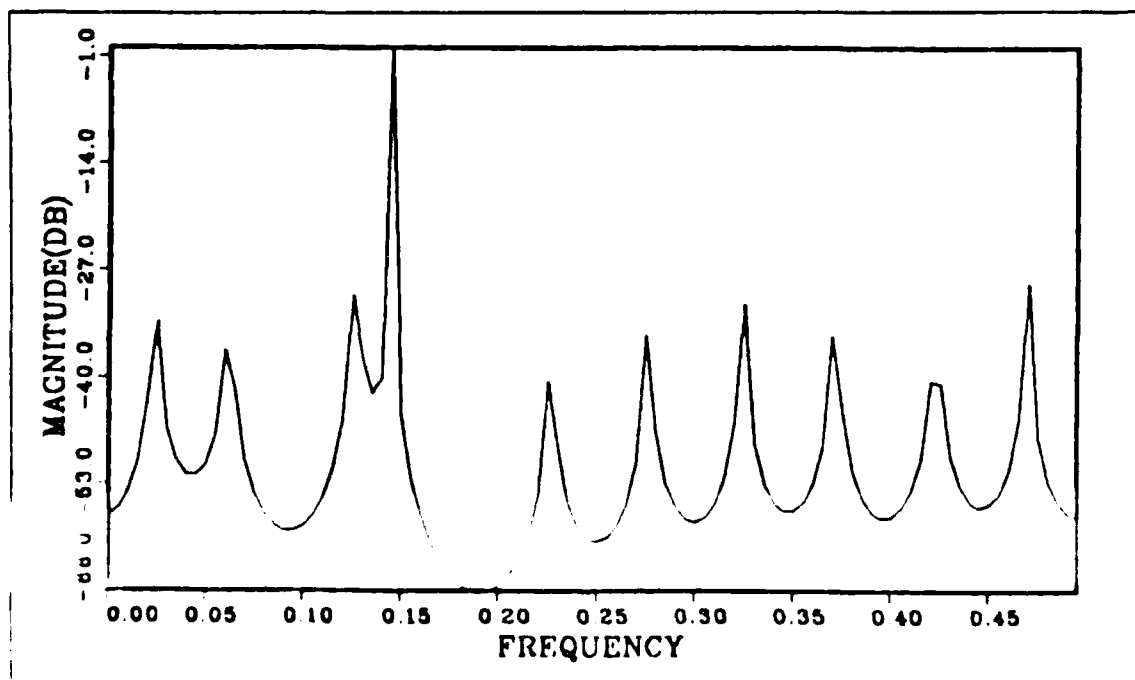


Figure 5.16 Frequency Spectrum (1 sinusoid, $M = 20$, $\text{SNR} = 30\text{dB}$).

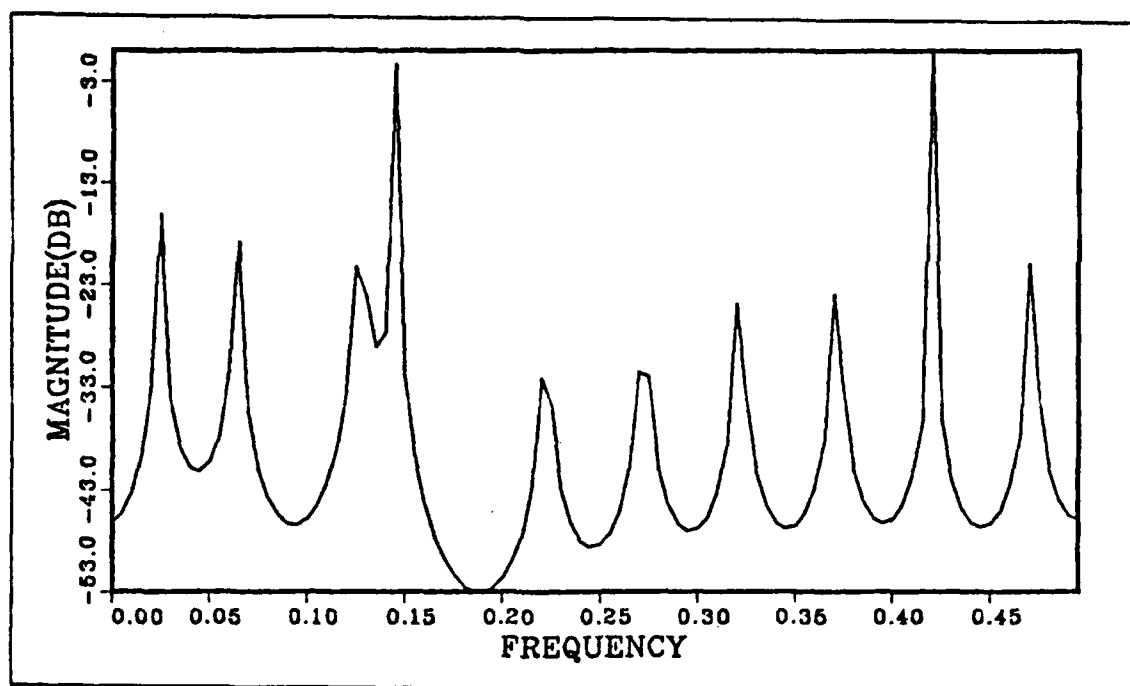


Figure 5.17 Frequency Spectrum (1 sinusoid, $M = 20$, $\text{SNR} = 20\text{dB}$).

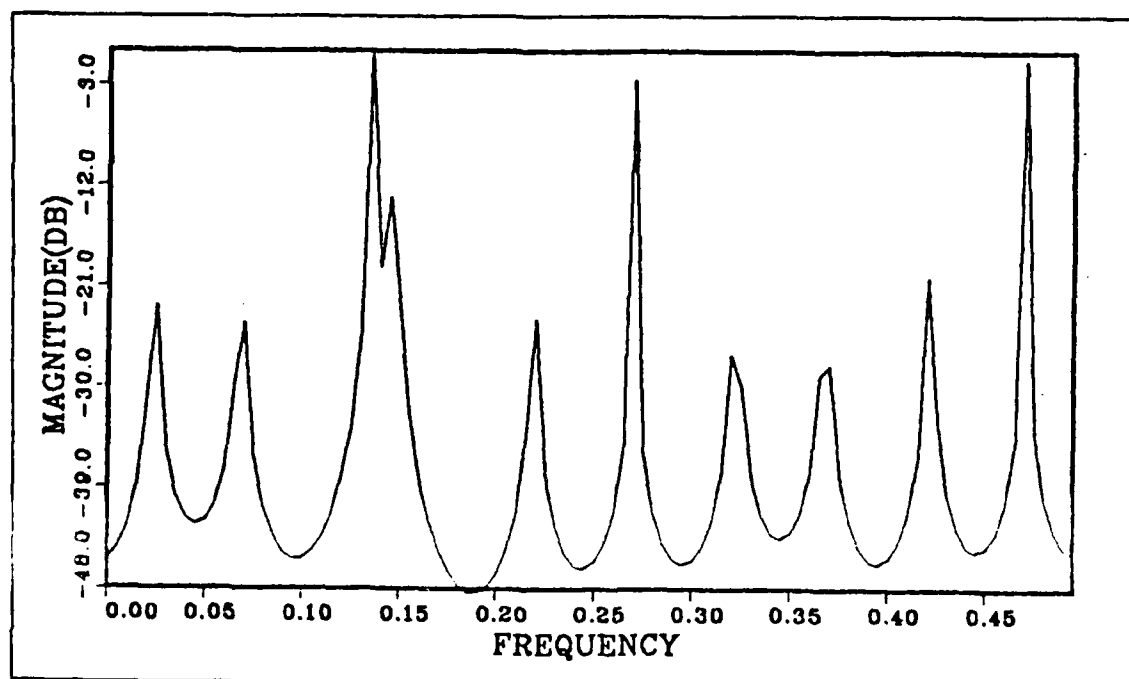


Figure 5.18 Frequency Spectrum (1 sinusoid, $M = 20$, $\text{SNR} = 10\text{dB}$).

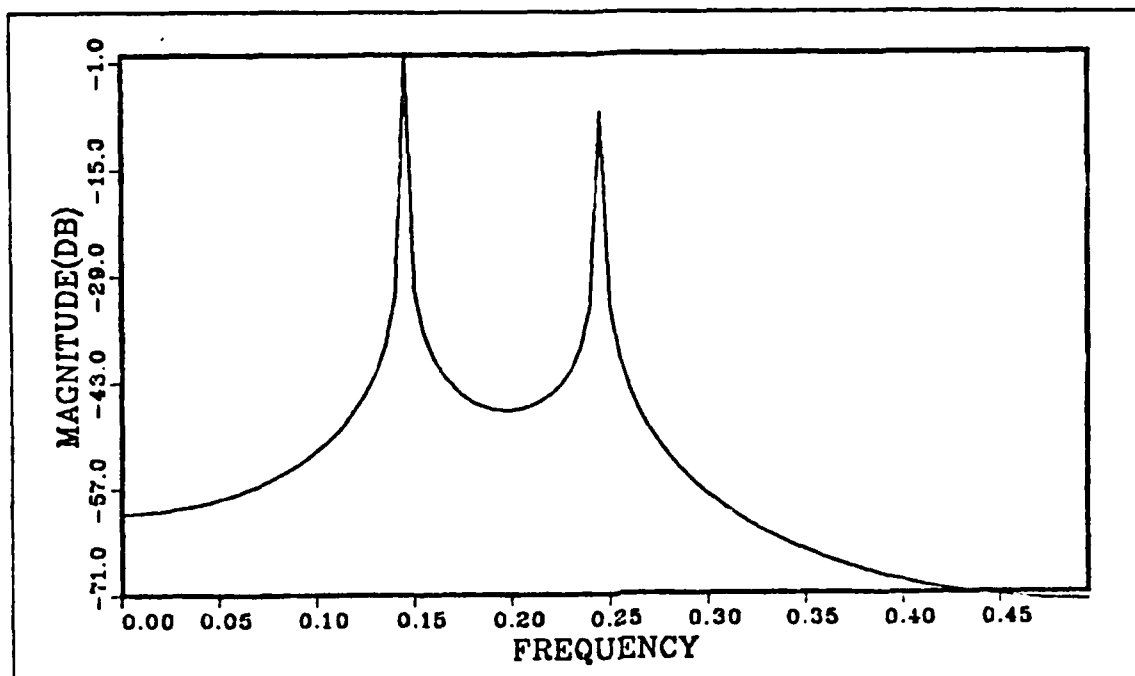


Figure 5.19 Frequency Spectrum (2 sinusoids, $M = 4$, $\mu = 0.02$).

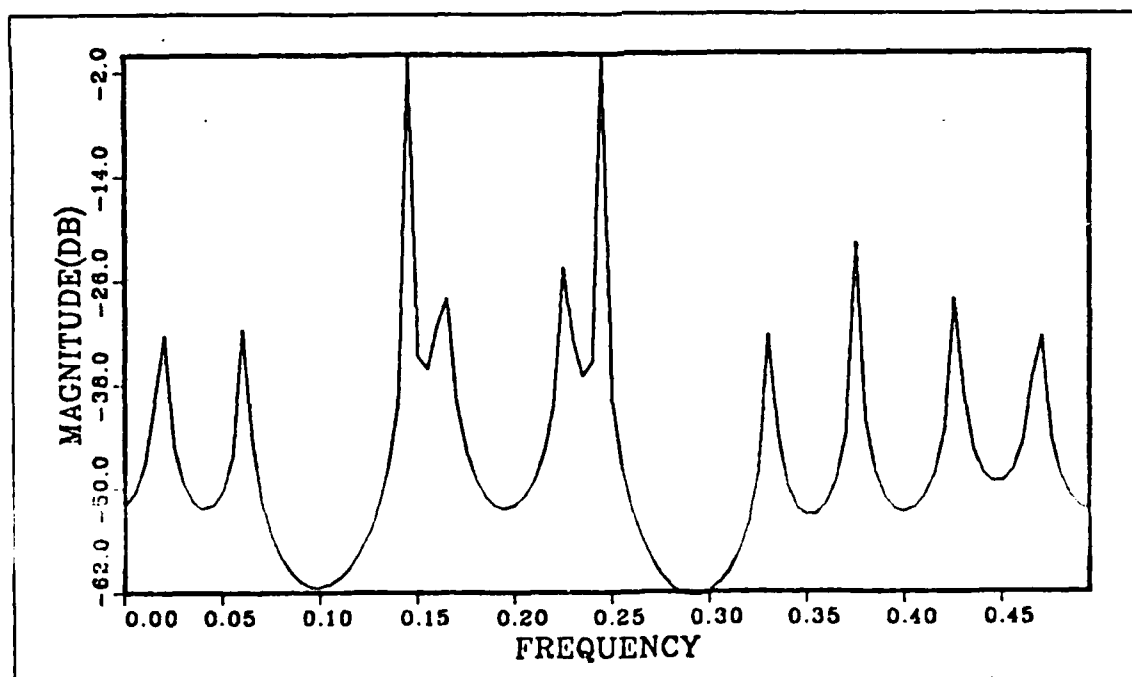


Figure 5.20 Frequency Spectrum (2 sinusoids, $M = 20$, $\mu = 0.022$).

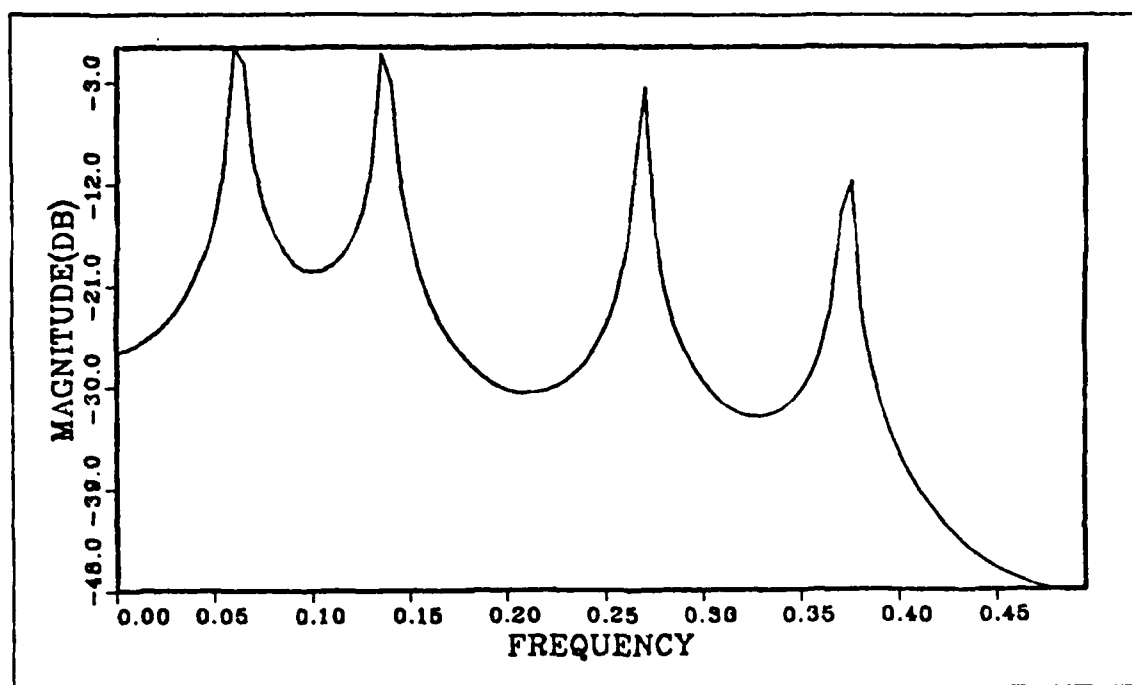


Figure 5.21 Frequency Spectrum (4 sinusoids, $M = 8$, $\mu = 0.015$).

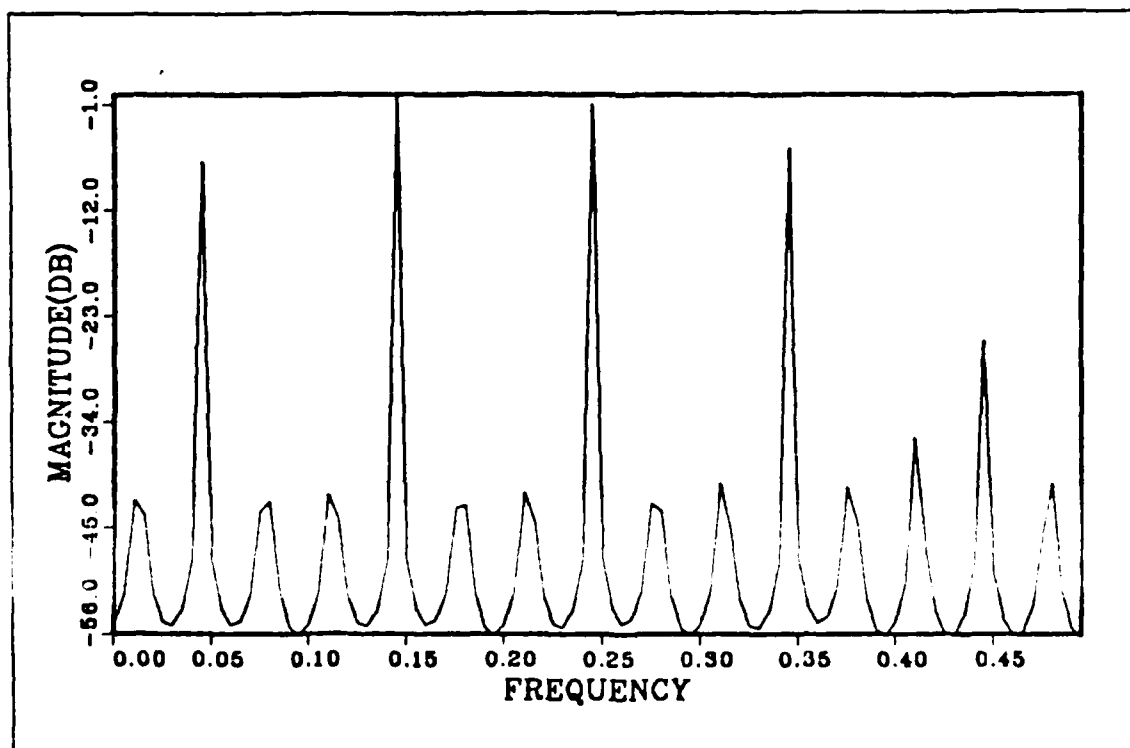


Figure 5.22 Frequency Spectrum (4 sinusoids, $M = 30$, $\mu = 0.014$).

E. SUMMARY, CONCLUSION AND SUGGESTED FUTURE WORK

The thesis investigates the application of lattice structures in Prony's method of spectral line estimation. The complete solution to Prony's method consists of three steps: (i) representing a given process of M sinusoids in noise in terms of complex exponentials, (ii) finding roots of a symmetric polynomial, and (iii) estimating the frequency, phase and amplitude information. In this thesis, however, we have emphasized only the frequency estimation problem.

The underlying theory of Prony's method has been briefly reviewed in Chapter II. By using a modified Prony's approach, we observed that the frequency estimation (as part of Prony's method) requires a polynomial with a linear phase property. The basics of the linear phase FIR filter and the lattice structure have been included in Chapter III. Also we have shown that a linear phase FIR transfer function can be realized from an FIR lattice using D number of additional unit delays and an adder, where D is determined by the order of the linear phase polynomial.

The principles of adaptive filtering and the LMS algorithm have been briefly studied in Chapter IV. We have derived a new LMS algorithm for the FIR non-linear phase and linear phase transfer functions in Chapter V. The problem spectral estimation using the new adaptive algorithm has been addressed, and the simulation results are included.

The significant outcome of the work is the derivation of an LMS type algorithm for a linear phase lattice structure and its application to spectral line estimation as part of Prony's method. As has been shown, the new algorithm yielded faster convergence rate compared to previously reported approximate algorithms. The application of this algorithm to spectral estimation produced high spectral resolution as illustrated by the simulation results.

However, we have observed spurious spectral responses, when the model order is higher than the required. Also we have observed that the SNR performance of the algorithm needs to be improved. For input SNRs less than about 10 dB, the algorithm performed poorly. Finally, we have dealt with only the frequency estimation part of Prony's method. In a future work, some of the above mentioned problems may be taken up.

APPENDIX A

EXAMPLES OF OBTAINING A LATTICE STRUCTURE FOR THE FIR TRANSFER FUNCTION

Example A.1

Obtaining a lattice structure for the general FIR transfer function. The unit sample response of a FIR transfer function is given by

$$H_3(z) = 0.5 + 0.25z^{-1} + 0.125z^{-2} + 0.0625z^{-3}$$

Solution: Given unit sample response is

$$H_3(z) = 0.5 + 0.25z^{-1} + 0.125z^{-2} + 0.0625z^{-3} \quad (A.1)$$

Using Eq.(3.22), we have polynomial for 3 sections

$$H_3(z) = b_{3,0} + b_{3,1}z^{-1} + b_{3,2}z^{-2} + b_{3,3}z^{-3} \quad (A.2)$$

Comparing Eqs.(A.1) and (A.2), we have

$$b_{3,0} = 0.5$$

$$b_{3,1} = 0.25$$

(A.3)

$$b_{3,2} = 0.125$$

$$b_{3,3} = 0.0625$$

Starting with $m = 3$ we have from Eq.(3.46)

$$k_3 = b_{3,3} = 0.0625$$

(A.4)

$$s_3 = b_{3,0} = 0.5$$

Now, we need to generate the coefficients for $H_2(z)$ and from Eq.(3.46)

$$b_{m-1,i} = \frac{s_m b_{m,i} - k_m b_{m,m-i}}{s_m^2 - k_m^2} \quad (\text{A.5})$$

And for $m=3$ and $i=0$ we have

$$b_{2,0} = \frac{s_3 b_{3,0} - k_3 b_{3,3}}{s_3^2 - k_3^2}$$

$$b_{2,0} = \frac{(0.5)(0.5) - (0.0625)(0.0625)}{(0.5)^2 - (0.0625)^2} = 1 \quad (\text{A.6})$$

Next, for $m=3$ and $i=1$ we get

$$b_{2,1} = \frac{s_3 b_{3,1} - k_3 b_{3,2}}{s_3^2 - k_3^2}$$

$$b_{2,1} = \frac{(0.5)(0.25) - (0.0625)(0.125)}{(0.5)^2 - (0.0625)^2} = 0.47619 \quad (\text{A.7})$$

and finally

$$b_{2,2} = \frac{s_3 b_{3,2} - k_3 b_{3,1}}{s_3^2 - k_3^2}$$

$$b_{2,2} = \frac{(0.5)(0.125) - (0.0625)(0.25)}{(0.5)^2 - (0.0625)^2} = 0.19048 \quad (\text{A.8})$$

from Eq.(3.46), we have

$$k_2 = b_{2,2} = 0.19048 \quad (\text{A.9})$$

$$s_2 = b_{2,0} = 1$$

The new polynomial is

$$H_2(z) = b_{2,0} + b_{2,1}z^{-1} + b_{2,2}z^{-2}$$

$$H_2(z) = 1 + 0.47619z^{-1} + 0.19048z^{-2} \quad (\text{A.10})$$

for $m=2$ and $i=0$, we have

$$b_{1,1} = \frac{b_{2,0} - k_2 b_{2,2}}{1 - k_2^2}$$

$$b_{1,1} = \frac{1 - (0.19048)(0.19048)}{1 - (0.19048)^2} = 1 \quad (\text{A.11})$$

and finally

$$b_{1,1} = \frac{b_{2,1} - k_2 b_{2,1}}{1 - k_2^2}$$

$$b_{1,1} = \frac{0.47619 - (0.19048)(0.47619)}{1 - (0.19048)^2} = 0.40000 \quad (\text{A.12})$$

From Eq.(3.46), we have

$$k_1 = b_{1,1} = 0.40000$$

$$s_1 = b_{1,0} = 1 \quad (\text{A.13})$$

Therefore, reflection coefficients and s coefficients of the FIR lattice structure are

$$\begin{aligned} k_1 &= 0.40000 \\ k_2 &= 0.19048 \\ k_3 &= 0.0625 \\ s_1 &= 1 \\ s_2 &= 1 \\ s_3 &= 0.5 \end{aligned}$$

Example A.2 (N = 4: even)

Obtaining a lattice structure for the linear phase FIR transfer function. A linear phase FIR transfer function is

$$H_3(z) = 0.154 + 0.462z^{-1} + 0.462z^{-2} + 0.154z^{-3} \quad (\text{A.14})$$

Solution: Eq.(A.14) can be written in the form of Eq.(3.35)

$$H_3(z) = a_0 + a_1z^{-1} + z^{-2}(a_1 + a_0z^{-1}) \quad (\text{A.15})$$

Comparing Eqs.(A.14) and (A.15), we get the following relationships.

$$a_0 = 0.154 \quad (\text{A.16})$$

$$a_1 = 0.462$$

To determine the lattice reflection coefficients of the corresponding linear phase FIR lattice structure and the number of unit delay, we use Eq.(3.29) which is

$$H_{N-1}(z) = F_M(z) + z^{-D} G_M(z) \quad (\text{A.17})$$

From Eq.(A.14), we have $N=4$ (even), the order of the polynomials $F_M(z)$ and $G_M(z)$, $M = (N/2)-1 = (4/2)-1 = 1$, and the number of unit delays, $D = N/2 = 2$. Now, from Eq.(3.37), we may have the forward prediction error transfer function as follows:

$$\begin{aligned} F_1(z) &= a_0 + a_1 z^{-1} \\ &= 0.154 + 0.462 z^{-1} \end{aligned} \quad (\text{A.18})$$

for $M=1$, Eq.(A.18) can be rewritten as

$$F_1(z) = b_{1,0} + b_{1,1} z^{-1} \quad (\text{A.19})$$

From Eqs.(A.18) and (A.19), we get the desired reflection coefficient and s coefficient of the linear phase FIR lattice structure as follows:

$$\begin{aligned} k_1 &= b_{1,1} = 0.462 \\ s_1 &= b_{1,0} = 0.154 \end{aligned}$$

Example A.3 (N=5:odd)

Obtaining a lattice structure for the linear phase FIR transfer function. A linear phase FIR transfer function is

$$H_4(z) = 0.15 - 0.45z^{-1} + 0.36z^{-2} - 0.45z^{-3} + 0.15z^{-4} \quad (\text{A.20})$$

Solution: Eq.(A.20) can be written in the form of Eq.(3.30)

$$H_4(z) = a_0 + a_1 z^{-1} + (1/2)a_2 z^{-2} + z^{-2} \{(1/2)a_2 + a_1 z^{-1} + a_0 z^{-2}\} \quad (\text{A.21})$$

Comparing Eqs.(A.20) and (A.21), we get the following relationships.

$$\begin{aligned} a_0 &= 0.15 \\ a_1 &= -0.45 \\ a_2 &= 0.36 \end{aligned} \quad (\text{A.22})$$

To determine the lattice reflection coefficients of the corresponding linear phase FIR lattice structure and the number of unit delay, we use Eq.(3.29) which is

$$H_{N-1}(z) = F_M(z) + z^{-D} G_M(z) \quad (\text{A.23})$$

From Eq.(A.20), we have $N=5(\text{odd})$, the order of the polynomials $F_M(z)$ and $G_M(z)$, $M = (N-1)/2 = (5-1)/2 = 2$, and the number of unit delays, $D = (N-1)/2 = 2$. Now, from Eq.(3.32), we may have the forward prediction error transfer function as follows:

$$\begin{aligned} F_2(z) &= a_0 + a_1 z^{-1} + (1/2) a_2 z^{-2} \\ &= 0.15 - 0.45 z^{-1} + 0.18 z^{-2} \end{aligned} \quad (\text{A.24})$$

for $M=2$, Eq.(A.24) can be rewritten as

$$F_2(z) = b_{2,0} + b_{2,1} z^{-1} + b_{2,2} z^{-2} \quad (\text{A.25})$$

From Eqs.(A.24) and (A.25), we have

$$k_2 = b_{2,2} = 0.18$$

$$s_2 = b_{2,0} = 0.15$$

Now, we need to generate the coefficients for $F_1(z)$ and from Eq.(3.46)

$$b_{1,0} = \frac{s_2 b_{2,0} - k_2 b_{2,2}}{s_2^2 - k_2^2} = \frac{(0.15)(0.15) - (0.18)(0.18)}{0.15^2 - 0.18^2} = 1 \quad (\text{A.26})$$

$$b_{1,1} = \frac{s_2 b_{2,1} - k_2 b_{2,1}}{s_2^2 - k_2^2} = \frac{(0.15)(-0.45) - (0.18)(-0.45)}{0.15^2 - 0.18^2} = -1.36364 \quad (\text{A.27})$$

From Eqs.(A.26) and (A.27), we have

$$k_1 = b_{1,1} = -1.36364$$

$$s_1 = b_{1,0} = 1$$

Therefore, reflection coefficients and s coefficients of the linear phase FIR lattice structure are

$$k_1 = -1.36364$$

$$k_2 = 0.18$$

$$s_1 = 1$$

$$s_2 = 0.15$$

APPENDIX B

COMPUTER PROGRAMS

```

*****Program 1*****
* This program is designed for the system identification experiment *
* which is shown in Section (IV.D). The learning curves can be obtained*
* by plotting the error, E, versus the update iteration, k. *
*****
*
*           Variable Definition
*
*   ISEED : seed for the random number generation (white noise)
*   AMU    : adaptation constant
*   k      : time index
*   M      : order of the FIR transfer function or total number of
*            lattice sections
*   NB     : number of iterations
*   A(I)   : coefficients of the FIR transfer function
*   B(I)   : reflection coefficients of the lattice
*   F(I)   : forward prediction error
*   G(I)   : backward prediction error
*   GD(I)  : delayed backward prediction error
*   SGL(I) : estimations of power
*   W(K)   : input of both FIR and lattice
*   YF(K)  : output of the FIR filter
*   YL(K)  : output of the lattice filter
*   ER(K)  : squared error
*
*           Variable Declaration
*
*   INTEGER ISEED,K,I,J,N,M,NB
*   REAL A(100),B(100),F(100),G(100),GD(100),YF(10000),YL(10000)
*   REAL X(100),SGL(100),ER(10000),W(10000),Y(10000),AMU,E
*
*           Set Adaptation Constant  $\mu$  and Number of Iterations NB
*
*   WRITE (5,1)
1  FORMAT(5X,'AMU',5X,'NB')
   READ(5,*) AMU,NB
*
*           Initialization
*
*   DO 10 K=1,100
*     A(K)=C

```

```

        B(K)=0
        X(K)=0
        F(K)=0
        G(K)=0
        GD(K)=0
        SGL(k)=1.0
10    continue
    DO 15 K=1,10000
        YF(K)=0
        YL(K)=0
        ER(K)=0
        W(K)=0
        Y(k)=0
15    CONTINUE
    E=0.
    R=1
    ISEED=343169
    M=2
    A(1) = 1.0
    A(2) = -0.89
    A(3) = +0.25
*
*           Random Number Generation
*           (mean zero, unit variance, white sequence)
*
    DO 20 N=1,NB
        CALL LNORM(ISEED,RN,1,1,0)
        W(N) = RN
20    CONTINUE
*
*           FIR Filter Calculation
*
    DO 30 K = 1,NB
        X(1)=W(K)
        YF(K)= A(1)*X(1)
        DO 40 I=1,M
            YF(K)=YF(K)+A(I+1)*X(I+1)
40        CONTINUE
        DO 45 I=1,M
            X(M+2-I)=X(M+1-I)
45        CONTINUE
30    CONTINUE
*
*           Lattice Filter Calculation
*
    DO 50 K =1,NB
        F(1)=W(K)

```

```

      G(1)=W(K)
      DO 60 I = 1,M
        F(I+1)=F(I)+B(I)*GD(I)
        G(I+1)=GD(I)+B(I)*F(I)
60    CONTINUE
      YL(K)=F(M+1)

      *
      *      Calculating the Error
      *
      E = YF(K) - YL(K)

      *
      *      Updating the Reflection Coefficients
      *
      CALL LMS (B,GD,E,AMU,SGL,M)

      *
      ER(K)=E**2
      DO 70 J=1,M
        GD(J)=G(J)
70    CONTINUE
      IF (K.EQ.R) THEN
        WRITE(6,300) K, B(1) B/2 ER(K)
        R=R+50
      END IF
      Y(k)=E
50  CONTINUE
300  FORMAT(3X,I6,5X,3(F10.7,4X))

      *
      *      Plotting the Learning Curve
      *
      CALL PLOT(Y,N)
      STOP
      END

      *
      *
      *
      SUBROUTINE LMS(B,GD,E,AMU,SGL,M)
      REAL B(100),GD(100),SGL(100),E,AMU
      INTEGER M
      DO 200 I=1,M
        SGL(I)=0.9*SGL(I)+0.1*(GD(I)*GD(I),
200    CONTINUE
      DO 210 I=1,M
        B(I)=B(I)+(AMU/SGL(I))*E*GD(I)
210    CONTINUE
      RETURN
      END

```

★
★

```

SUBROUTINE PLOT(Y,N)
DIMENSION Y(N),X(10000)
ISTP=N/10
DO 10 J=1,N
10  X(J)=J
C  CALL TEK618
C  CALL PRTPLT(72,6)
  CALL SHERPA('PARKPARK','A',3)
  CALL PHY5OP(1,1,1)
  CALL RESET('ALL')
  CALL PAGE(8.5,11.0)
  CALL HWROT('AUTO')
  CALL XINTAX
  CALL AREA2D(5.0,2.8)
  CALL HEIGHT(0.10)
  CALL COMPLEX
  CALL SHDCHR(90.0,1,0.002,1)
  CALL HEADIN('LEARNING CURVES',100,2.0,1)
  CALL XNAME('ITERATIONS',100)
  CALL YNAME('ERRORS',100)
  CALL MESSAG('ADAPTIVE LATTICE(AMU=.5,FIG4.8)$',100,3.0,-0.8)
  CALL FRAME
  CALL GRAF(0,ISTP,N,-3.0,1.5,3.0)
C  CALL THKCRV(0.02)
  CALL CURVE(X,Y,N,0)
  CALL ENDPL(0)
  CALL DONEPL
  RETURN
END

```

```

*****Program 2*****
* This program is designed for the system identification experiment *
* using the LMS algorithm which was derived in Section (V.B). *
*****
*
*
*
      INTEGER ISEED,K,I,J,N,M,NB
      REAL A(100),B(100),F(100),G(100),GD(100),X(100),YF(5000),YL(5000)
      REAL ER(5000),W(5000),SGL(100),PH(100,100),PS(100,100)
      REAL PSD(100,100),GR(100),Y(5000),AMU,E
*
*      Set Adaptation Constant  $\mu$  and Number of Iterations NB
*
      WRITE(5,1)
1      FORMAT(5X,'AMU',5X,'NB')
      READ(5,*) AMU,NB
*
*      Initialization
*
      E=0.
      R=1
      DO 5 K= 1,5000
          W(K)=0
          YF(K)=0
          YL(K)=0
          ER(K)=0
          Y(K)=0
5      CONTINUE
          DO 10 K=1,100
              A(K)=0
              B(K)=0
              X(K)=0
              F(K)=0
              G(K)=0
              GD(K)=0
              SGL(K)=1.0
              GR(K)=0
10      CONTINUE
              DO 15 K=1,100
                  DO 15 L=1,100
                      PH(K,L)=0
                      PS(K,L)=0
                      PSD(K,L)=0
15      CONTINUE
      ISEED=343169
      M=2

```



```

A(1) = 1.0
A(2) = -0.89
A(3) = +0.25

*
*      Random Number Generation
*      (mean zero, unit variance, white sequence)
*

DO 20 N=1,NB
  CALL LNORM(ISEED,RN,1,1,0)
  W(N) = RN
20 CONTINUE

*
*      FIR Filter
*

DO 30 K = 1,NB
  X(1)=W(K)
  YF(K)= A(1)*X(1)
  DO 40 I=1,M
    YF(K)=YF(K)+A(I+1)*X(I+1)
40 CONTINUE
  DO 45 I=1,M
    X(M+2-I)=X(M+1-I)
45 CONTINUE
30 CONTINUE

*
*      Lattice Filter
*

DO 50 K =1,NB
  F(1)=W(K)
  G(1)=W(K)
  DO 60 I = 1,M
    F(I+1)=F(I)+B(I)*GD(I)
    G(I+1)=GD(I)+B(I)*F(I)
60 CONTINUE
  YL(K)=F(M+1)
  E = YF(K) - YL(K)

*
*      Updating the Reflection Coefficients
*

CALL MLMS (PH,PS,PSD,GR,F,G,GD,B,SGL,E,AMU,M)
ER(K)=E**2
DO 70 J=1,M
  GD(J)=G(J)
70 CONTINUE
IF (K.EQ.R) THEN
  WRITE(6,300) K, B(1),B(2),B(3),B(4),B(5),B(6),B(7),B(8),E
  R=R+30

```

```

        END IF
        Y(K)=E
c      WRITE(6,100) K,W(K),YF(K),YL(K),E,ER(K)
50      CONTINUE
300     FORMAT(1X,I6,2X,9(F10.7,2X))
100     FORMAT(3X,I3,2X,5(F10.7,2X))
*
*           Plotting the Learning Curve
*
      CALL PLOT(Y,N)
      STOP
      END
*
*
*
      SUBROUTINE MLMS(PH,PS,PSD,GR,F,B,BD,R,SGL,EK,AMU,N)
      DIMENSION PH(100,100),PS(100,100),PSD(100,100),F(100),B(100)
1, BD(100),R(100),GR(100),SGL(101)
      GR(N)=BD(N)
      DO 200 I=2,N
      PH(I,1)=BD(N+1-I)
      PS(I,1)=F(N+1-I)
      IT=I-1
      DO 10 K=1,IT
      PH(I,K+1)=PH(I,K)+R(N+1-I+K)*PSD(I,K)
10     PS(I,K+1)=PSD(I,K)+R(N+1-I+K)*PH(I,K)
      DO 20 K=1,IT
      20     PSD(I,K)=PS(I,K)
200     CONTINUE
      DO 210 K=2,N
      210     GR(N+1-K)=PH(K,K)
      DO 211 K=1,N
      211     SGL(K)=.90*SGL(K)+.10*GR(K)*GR(K)+1.0
      DO 220 I=1,N
      R(I)=R(I)+(AMU/SGL(I))*EK*GR(I)
      IF(R(I).GE.1.0) R(I)=0.
      IF(R(I).LE.-1.0) R(I)=0.
220     CONTINUE
      RETURN
      END
*
*
*
      SUBROUTINE PLOT(Y,N)
      DIMENSION Y(N),X(5000)
      ISTP=N/10
      DO 10 J=1,N

```

```

10  X(J)=J
c   CALL TEK618
c   CALL PRTPLT(72,6)
    CALL SHERPA('PARKPARK','A',3)
    CALL PHYSOR(1.,1.)
    CALL RESET('ALL')
    CALL PAGE(3.5,11.0)
    CALL HWROT('AUTO')
    CALL XINTAX
    CALL AREA2D(5.0,2.8)
    CALL HEIGHT(0.10)
    CALL COMPLX
    CALL SHDCHR(90.0,1,0.002,1)
    CALL HEADIN('LEARNING CURVES',100,2.0,1)
    CALL XNAME('ITERATIONSS',100)
    CALL YNAME('ERRORS',100)
    CALL MESSAG('ADAPTIVE LATTICE(AMU=.5,FIG5.5)S',100,3.0,-0.8)
    CALL FRAME
    CALL GRAF(0.1STP,N,-2.5,1.25,2.5)
c   CALL THKCRV(0.02)
    CALL CURVE(X,Y,N,0)
    CALL ENDPL(0)
    CALL DONEPL
    RETURN
    END

```

```

*****Program 3*****
* This program is designed for the system identification experiment. *
* The LMS algorithm shown in Section (V.C) was extended to the linear *
* phase FIR lattice filter. *
*****
*
*
*
      INTEGER ISEED,K,I,J,N,M,NB,R,MA,ML
      REAL A(100),B(100),F(100),G(100),GD(100),YF(3000),YL(3000)
      REAL X(100),H(100),YD(3000),ER(3000),W(3000),YFF(3000),YLL(3000)
      REAL SGL(100),PH(100,100),PS(100,100),PSD(100,100)
      REAL GR(100),GRD(100,100),GRB(100),Y(3000),AMU,E,C

*
*      Set Adaptation Constant  $\mu$  and Number of Iterations NB
*
      WRITE(5,1)
1    FORMAT(5X,'AMU',5X,'NB')
      READ(5,*) AMU,NB

*
*      Initialization
*
      E=0.
      R=1
      M=4
      ISEED=343169
      DO 10 K=1,100
          A(K)=0
          B(K)=0
          X(K)=0
          F(K)=0
          G(K)=0
          GD(K)=0
          H(K)=0
          SGL(K)=1.0
          GR(K)=0
          GRB(K)=0
10     CONTINUE
      DO 11 K=1,3000
          W(K)=0
          YF(K)=0
          YL(K)=0
          YFF(K)=0
          YLL(K)=0
          YD(K)=0
          ER(K)=0
          Y(K)=0

```

```

11      CONTINUE
      DO 15 K=1,100
        DO 15 L=1,100
          PH(K,L)=0
          PS(K,L)=0
          PSD(K,L)=0
          GRD(K,L)=0
15      CONTINUE
c      C=.154
c      A(1)=.154/C
c      A(2)=.462/C
      C=.15
      A(1)=.15/C
      A(2)=-.45/C
      A(3)=.36/C
*
*      Random Number Generation
*      (mean zero, unit variance, white sequence)
*
      DO 20 N=1,NB
        CALL LNORM(ISEED,RN,1,1,0)
        W(N) = RN
20      CONTINUE
      IF (MOD(M,2).EQ.0) MA=M/2
      IF (MOD(M,2).NE.0) MA=(M+1)/2
      IF (MOD(M,2).EQ.0) ML=MA
      IF (MOD(M,2).NE.0) ML=MA-1
*
*      Separation of Coefficients
*
      IF (MOD(M,2).EQ.0) THEN
        DO 21 I=1,MA
          H(I)=A(I)
          H(MA+2+I)=A(MA+1-I)
21      CONTINUE
          H(MA+1)=A(MA+1)/2
          H(MA+2)=A(MA+1)/2
        END IF
      IF (MOD(M,2).NE.0) THEN
        DO 22 I=1,MA
          H(I)=A(I)
          H(MA+I)=A(MA+1-I)
22      CONTINUE
        END IF
*
*      LINEAR PHASE FIR FILTER
*

```

```

DO 30 K = 1,NB
  X(1)=W(K)
  YF(K)=H(1)*X(1)
  IF (MOD(M,2).EQ.0) THEN
    DO 40 I=1,MA
      YF(K)=YF(K)+H(I+1)*X(I+1)
40    CONTINUE
      YFF(K)=YF(K)
      DO 41 I=1,MA+1
        YFF(K)=YFF(K)+P(MA+1+I)*X(MA+I)
41    CONTINUE
      END IF
      IF (MOD(M,2).NE.0) THEN
        DO 42 I=1,MA-1
          YF(K)=YF(K)+H(I+1)*X(I+1)
42    CONTINUE
          YFF(K)=YF(K)
          DO 43 I=1,MA
            YFF(K)=YFF(K)+H(MA+I)*X(MA+I)
43    CONTINUE
          END IF
          DO 45 I=1,M
            X(M+2-I)=X(M+1-I)
45    CONTINUE
30  CONTINUE
*
*    LATTICE FILTER
*
DO 50 K =1,NB
  F(1)=W(K)
  G(1)=W(K)
  DO 60 I = 1,ML
    F(I+1)=F(I)+B(I)*GD(I)
    G(I+1)=GD(I)+B(I)*F(I)
60  CONTINUE
    YL(K)=F(ML+1)
    YD(K)=G(ML+1)
    YLL(K)=YL(K)+YD(K-MA)
    E = YFF(K) - YLL(K)
*
*    Updating the Reflection Coefficients
*
CALL MLMS (P,Q,QD,GRB,GRD,F,G,GD,B,SGL,E,AMU,ML,MA)
ER(K)=E**2
DO 70 J=1,ML
  GD(J)=G(J)
70  CONTINUE

```

```

        Y(K)=E
        WRITE(6,300) K,Y(K)
50      CONTINUE
*
*          Plotting the Learning Curve
*
        CALL PLOT(Y,N)
300     FORMAT(3X,I6,3X,F10.5)
        STOP
        END
*
*
*
        SUBROUTINE MLMS(PH,PS,PSD,GRB,GRD,F,B,BD,R,SGL,EK,AMU,N,MA)
        DIMENSION PH(100,100),PS(100,100),PSD(100,100),GRD(100,100)
1       GRB(100),F(100),B(100),BD(100),R(100),GR(100),SGL(101)
        GR(N)=BD(N)
        GRB(N)=F(N)
        DO 200 I=2,N
        PH(I,1)=BD(N+1-I)
        PS(I,1)=F(N+1-I)
        IT=I-1
        DO 110 K=1,IT
        PH(I,K+1)=PH(I,K)+R(N+1-I+K)*PSD(I,K)
110     PS(I,K+1)=PSD(I,K)+R(N+1-I+K)*PH(I,K)
        DO 120 K=1,IT
        PSD(I,K)=PS(I,K)
120     PSD(I,K)=PS(I,K)
200     CONTINUE
        DO 210 K=2,N
        GR(N+1-K)=PH(K,K)
210     GRB(N+1-K)=PS(K,K)
        DO 220 K=1,N
        DO 220 L=1,MA
220     GRD(K,MA+2-L)=GRD(K,MA+1-L)
        DO 230 K=1,N
230     GRD(K,1)=GRB(K)
        DO 240 K=1,N
240     GRB(K)=GRD(K,MA+1)
        DO 250 K=1,N
250     GR(K)=GR(K)+GRB(K)
        DO 260 K=1,N
260     SGL(K)=.90*SGL(K)+.10*GR(K)*GR(K)+1.0
        DO 270 I=1,N
        R(I)=R(I)+(AMU/SGL(I))*EK*GR(I)
c       IF(R(I).GE.1.0) R(I)=0.
c       IF(R(I).LE.-1.0) R(I)=0.
270     CONTINUE

```

```

      RETURN
      END

*
*
*
      SUBROUTINE PLOT(Y,N)
      DIMENSION Y(N),X(3000)
      ISTP=N/10
      DO 10 J=1,N
10    X(J)=J
      c    CALL TEK618
      c    CALL PRTPLT(72,6)
      CALL SHERPA('PARKPARK','A',3)
      CALL PHYSOR(1.,1.)
      CALL RESET('ALL')
      CALL PAGE(8.5,11.)
      CALL HWROT('AUTO')
      CALL XINTAX
      CALL AREA2D(5.0,2.8)
      CALL HEIGHT(0.10)
      CALL COMPLX
      CALL SHDCHR(90.0,1,0.002,1)
      CALL HEADIN('LEARNING CURVES',100,2.0,1)
      CALL XNAME('ITERATIONSS',100)
      CALL YNAME('ERRORS',100)
      CALL MESSAG('ADAPTIVE G-M LATTICE(F5.10,AMU=0.1)$',100,3.0,-0.8)
      CALL FRAME
      CALL GRAF(0,ISTP,N,-8.0,4.0,8.0)
      c    CALL THKCRV(0.02)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      CALL DONEPL
      RETURN
      END

```



```

*****Program 4*****
* This program is designed for the estimation of spectral lines in
* white noise. The input process x(k) consists of a signal in noise.
* and a signal may be a single, or multiple sinusoids. The algorithm
* was derived in Section (V.D).
*****
*
*
*
      INTEGER ISEED,K,I,J,N,M,NB,R,MA,ML,D
      REAL A(100),B(100),F(100),G(100),GD(100),YL(5000)
      REAL Y(100),YD(5000),W(5000),YLL(5000),YD(5000),INP,5000
      REAL SGL(101),PH(100,100),PS(100,100),PSD(100,100)
      REAL GRD(100,100),GRB(100),GR(100)
      REAL RE(100),IM(100),AJ(100),SD,SNR,AVG,AMP,AMU,E
*
*      Set Adaptation Constant  $\mu$  and Number of Iterations NB
*
      WRITE(5,1)
      FORMAT(5X,'AMU',5X,'NB')
      READ(5,*) AMU,NB
*
*      Initialization
*
c      ISPEC=1000
      SNR=30.
      SD=10**(-(SNR/20))
      AMP=SQRT(2.)
      AVG=0.
      E=0.
c      R=1
      PI=3.141592654
      M=8
      MP1=M+1
      ISEED=343169
      IF (MOD(M,2).EQ.0) THEN
      MA=M/2
      ML=MA
      ELSE
      MA=(M+1)/2
      ML=MA-1
      END IF
      DO 10 K=1,100
          A(K)=0
          B(K)=0
          F(K)=0
          G(K)=0

```

```

        GD(K)=0
        SGL(K)=1.0
        GR(K)=0
        GRB(K)=0
        RE(K)=0
        IM(K)=0
        AJ(K)=0
        Y(K)=0
10      CONTINUE
        DO 11 K=1,5000
            W(K)=0
            YL(K)=0
            YLL(K)=0
            YD(K)=0
            ER(K)=0
            INP(K)=0
11      CONTINUE
        DO 15 K=1,100
            DO 15 L=1,100
                PH(K,L)=0
                PS(K,L)=0
                PSD(K,L)=0
                GRD(K,L)=0
15      CONTINUE
*
*          Random Number Generation
*          (mean zero, unit variance, white sequence)
*
        DO 20 N=1,NB
            CALL LNORM(ISEED,RN,1,1,0)
            W(N) = SD*RN+AVG
20      CONTINUE
*
*          LATTICE FILTER
*
        DO 50 K =1,NB
            AK=K-1
            INP(K)=AMP*COS(2*PI*.15*AK)+W(K)
            INP(K)=AMP*(COS(2*PI*.15*AK)+COS(2*PI*.25*AK) -
            INP(K)=AMP*(COS(2*PI*.05*AK)+COS(2*PI*.15*AK) -
            *)+COS(2*PI*.35*AK))+W(K)
            F(1)=INP(K)
            G(1)=INP(K)
            DO 60 I = 1,ML
                F(I+1)=F(I)+B(I)*GD(I)
                G(I+1)=GD(I)+B(I)*F(I)
60      CONTINUE

```

AD-A184 978

AN ADAPTIVE LATTICE ALGORITHM FOR SPECTRAL LINE
ESTIMATION(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA
I K PARK JUN 87

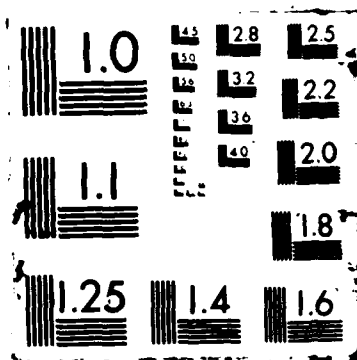
2/2

UNCLASSIFIED

F/G 12/1

NL

END
001
010



```

        YL(K)=F(ML+1)
        YD(K)=G(ML+1)
        YLL(K)=YL(K)+YD(K-MA)
        E = YLL(K)
*
*       Updating the Reflection Coefficients
*
        CALL MLMS (PH,PS,PSD,GR,GRB,GRD,F,G,GD,B,SGL,E,AMU,ML,MA)
        ER(K)=E**2
        DO 70 J=1,ML
            GD(J)=G(J)
70      CONTINUE
        IF (K.NE.ISPEC) GO TO 50
        IF (K.EQ.NB) THEN
        C      WRITE(6,300) K,INP(K),E
        C      WRITE(6,301) K,B(1),B(2),B(3),B(4),B(5),B(6),B(7),B(8),B(9)
        C      R=R+100
*
*       Determine the FIR Coefficients from the Lattice Reflection
*       Coefficients
*
        CALL STEPUP (A,B,ML)
        IF (MOD(M,2).EQ.0) THEN
            DO 80 I=1,M/2
80          A(M+2-I)=A(I)
            A(M/2+1)=2*A(M/2+1)
        ELSE
            DO 81 I=1,(M+1)/2
81          A(M+2-I)=A(I)
        END IF
        C      WRITE (6,600) (I,A(I),I=1,MP1)
*
*       Calculate the Power Spectrum
*
        CALL SPEC (RE,IM,A,AJ,M,PI)
        D=100
        C      WRITE (6,601) (.005*Z,AJ(Z),Z=1,D)
*
*       Plotting the Spectrum
*
        CALL PLOT(AJ,D)
        C      ISPEC=ISPEC+1000
        END IF
50      CONTINUE
        C      WRITE (6,300) (YLL(K),K=1,NB)
300     FORMAT(10(F10.7,1X))
301     FORMAT(1X,I5,1X,10(F10.7,1X))

```

```

600  FORMAT (1X,I5,5X,F15.7)
601  FORMAT (1X,F5.3,5X,F15.7)
STOP
END

```

```

*
*
*

```

```

SUBROUTINE MLMS(PH,PS,PSD,GR,GRB,GRD,F,B,BD,R,SGL,EK,AMU,N,MA)
DIMENSION PH(100,100),PS(100,100),PSD(100,100),GR(100),GRB(100)
1,GRD(100,100),F(100),B(100),BD(100),R(100),SGL(101)
GR(N)=BD(N)
GRB(N)=F(N)
DO 200 I=2,N
PH(I,1)=BD(N+1-I)
PS(I,1)=F(N+1-I)
IT=I-1
DO 110 K=1,IT
PH(I,K+1)=PH(I,K)+R(N+1-I+K)*PSD(I,K)
PS(I,K+1)=PSD(I,K)+R(N+1-I+K)*PH(I,K)
110 CONTINUE
DO 120 K=1,IT
120 PSD(I,K)=PS(I,K)
200 CONTINUE
DO 210 K=2,N
GR(N+1-K)=PH(K,K)
GRB(N+1-K)=PS(K,K)
210 CONTINUE
DO 220 K=1,N
DO 220 L=1,MA
220 GRD(K,MA+2-L)=GRD(K,MA+1-L)
DO 230 K=1,N
230 GRD(K,1)=GRB(K)
DO 240 K=1,N
240 GRB(K)=GRD(K,MA+1)
DO 250 K=1,N
250 GR(K)=GR(K)+GRB(K)
DO 260 K=1,N
260 SGL(K)=.90*SGL(K)+.10*GR(K)*GR(K)+1.0
DO 270 I=1,N
R(I)=R(I)-(AMU/SGL(I))*EK*GR(I)
IF(R(I).GE.1.0) R(I)=0.
IF(R(I).LE.-1.0) R(I)=0.
270 CONTINUE
RETURN
END

```

```

*
*

```

*

```

SUBROUTINE STEPUP (A,B,ML)
DIMENSION A(100),C(100),B(100)
A(1)=1.
A(2)=B(1)
DO 30 MINC=2,ML
DO 10 J=1,MINC
JB=MINC-J+1
10 C(J)=A(JB)
DO 20 IP=2,MINC
20 A(IP)=A(IP)+B(MINC)*C(IP-1)
A(MINC+1)=B(MINC)
30 CONTINUE
RETURN
END

```

*

*

*

```

SUBROUTINE SPEC(RE,IM,A,AJ,M,PI)
REAL RE(100),IM(100),A(100),AJ(100),Y(100)
RE(1)=A(1)
IM(1)=0.
DO 91 J=1,100
DO 92 I=1,M
RE(I+1)=RE(I)+A(I+1)*COS(2*I*PI*.5*J/100)
IM(I+1)=IM(I)+A(I+1)*SIN(2*I*PI*.5*J/100)
92 CONTINUE
AJ(J)=-10.*ALOG10(RE(M+1)*RE(M+1)+IM(M+1)*IM(M+1))
91 CONTINUE
TEMP=AJ(1)
DO 93 L=2,100
IF (AJ(L).GT.TEMP) TEMP=AJ(L)
93 CONTINUE
DO 94 L=1,100
AJ(L)=AJ(L)-TEMP
94 CONTINUE
RETURN
END

```

*

*

*

```

SUBROUTINE PLOT(Y,N)
DIMENSION Y(N),X(100)
C ISTD=N/10
C DO 10 J=1,N
C10 X(J)=J
DF=.5/N

```

```

      X(1)=0
      DO 10 K=2,N
10    X(K)=X(K-1)+DF
      XMIN=X(1)
      XMAX=X(N)
      XSTP=10*DF
      IYMIN=Y(1)
      IYMAX=Y(1)
      DO 20 K=2,N
      IF(Y(K).GT.IYMAX) IYMAX=Y(K)
      IF(Y(K).LT.IYMIN) IYMIN=Y(K)
20    CONTINUE
      IYSTP=(IYMAX-IYMIN)/5
c     CALL TEK618
c     CALL PRTPLT(72,6)
      CALL SHERPA('PARKPARK','A',3)
      CALL RESET('ALL')
      CALL PAGE(8.5,11.0)
      CALL HWROT('AUTO')
      CALL XINTAX
      CALL AREA2D(5.0,2.8)
      CALL HEIGHT(0.10)
      CALL COMPLX
      CALL SHDCHR(90.0,1,0.002,1)
      CALL HEADIN('FREQUENCY SPECTRUM$',100,2.0,1)
      CALL XNAME('FREQUENCY$',100)
      CALL YNAME('MAGNITUDE(DB)$',100)
      CALL MESSAG('FIGURE 5.21 (M=8 ,SNR=30DB)$',100,3.0,-0.8)
c     CALL MESSAG('MODEL ORDER SELECTION(4 SINUSOIDS)$',100,3.0,-0.8)
      CALL THKFRM(0.03)
      CALL FRAME
      CALL GRAF(XMIN,XSTP,XMAX,IYMIN,IYSTP,IYMAX)
c     CALL THKCRV(0.02)
      CALL CURVE(X,Y,N,0)
      CALL ENDPL(0)
      CALL DONEPL
      RETURN
      END

```


LIST OF REFERENCES

1. Markel, J. D. and A. H. Gray, Jr., *Linear Prediction of Speech*, Berlin, Springer Verlag, pp. 92-103, 1976.
2. Rabiner, L. R. and B. Gold, *Theory and Application of Digital Signal Processing*, Englewood Cliffs, New Jersey, Prentice-Hall, Inc., 1975.
3. Oppenheim, A. V. and R. W. Schaffer, *Digital Signal Processing*, Englewood Cliffs, New Jersey, Prentice-Hall, Inc., pp. 157-160, 1975.
4. Strum, R. D. and D. E. Kirk, *ECE 3400 Textbook*, Carmel, California, 1984.
5. Chen, C. T., *One-Dimensional Digital Signal Processing*, New York and Basel, Marcel Dekker, Inc., pp. 191-215, 1979.
6. Widrow, B. and S. D. Stearns, *Adaptive Signal Processing*, Englewood Cliffs, New Jersey, Prentice-Hall, Inc., pp. 99-114, 1985.
7. Honig, M. L. and D. G. Messerschmitt, *Adaptive Filters*, Kluwer Academic Publishers, pp. 85-123, 1984.
8. Haykin S., *Introduction to Adaptive Filters*, Macmillan Publishing Company, pp. 162-211, 1984.
9. Haykin S., *Nonlinear Method of Spectral Analysis*, Springer-Verlag, Berlin, Heidelberg, New York, pp. 41-48, 1979.
10. Marple, S. L., *Spectral Line Analysis by Pisarenko and Prony Methods*, 1979 IEEE Int. Conference on Acoust., Speech and Signal Processing, pp. 159-161.
11. Kay, S. M. and S. L. Marple, *Spectral Analysis - A Modern Perspective*, Proc IEEE, Vol. 69, pp. 1395-1414, Nov. 1981.
12. Friedlander, B. and M. Morf, *Least Squares Algorithms for Linear-Phase Filtering*, IEEE Trans. on Acoust., Speech and Signal Processing, Vol. ASSP-30, pp. 381-390, June 1982.
13. Marple, S. L., *Fast Algorithms for Linear Prediction and System Identification Filters with Linear Phase*, IEEE Trans. on Acoust., Speech and Signal Processing, Vol. ASSP-30, pp. 942-953, Dec. 1982.
14. Widrow, B., J. R. Glover, J. M. McCool, J. Kaunitz, C. S. Williams, R. H. Hearn, J. R. Zeidler, E. Dong and R. C. Goodlin, *Adaptive Noise Cancelling : Principles and Applications*, Proc IEEE, Vol. 63, No. 12, pp. 1692-1716, Dec. 1975.
15. Widrow, B., J. M. McCool and M. Ball, *The Complex LMS Algorithm*, Proc IEEE, pp. 719-720, Apr. 1975.
16. Widrow, B., J. M. McCool, M. G. Larimore and C. R. Johnson, *Stationary and Nonstationary Learning Characteristics of the LMS Adaptive Filter*, Proc IEEE, Vol. 64, No. 8, pp. 1151-1162, Aug. 1976.

17. Griffiths, L. J., *An Adaptive Lattice Structure for Noise Cancelling Applications*, 1978 IEEE Int. Conference on Acoust., Speech and Signal Processing, pp. 87-90.
18. Griffiths, L. J., *A Continuously-Adaptive Filter Implemented as a Lattice Structure*, 1977 IEEE Int. Conference on Acoust., Speech and Signal Processing, pp. 683-686.
19. Makhoul, J., *Spectral Linear Prediction : Properties and Applications*, IEEE Trans. on Acoust., Speech and Signal Processing, Vol. ASSP-23, No. 3, pp. 283-296, Jun. 1975.
20. Makhoul, J. and R. Viswanathan, *Adaptive Lattice Methods for Linear Prediction*, 1978 IEEE Int. Conference on Acoust., Speech and Signal Processing, pp. 83-86.
21. Makhoul, J., *Linear Prediction : A Tutorial Review*, Proc IEEE, Vol. 63, No. 4, pp. 561-578, Apr. 1975.
22. Tummala, M. and S. R. Parker, *A New Efficient Adaptive Cascade Lattice Structure*, Submitted to IEEE Transactions on Circuits and System, Special Issue on Adaptive Systems and Applications, Aug. 1986; revised Feb. 1987.
23. Satorius, E. H., and J. D. Pack, *Application of Least Squares Lattice Algorithms to Adaptive Equalization*, IEEE Transactions on Communications, Vol. COM-29, pp. 136-142, Feb. 1981.
24. Gibson, C. and S. Haykin, *Learning Characteristics of Adaptive Lattice Filtering Algorithms*, IEEE Trans. on Acoust., Speech and Signal Processing, Vol. ASSP-28, pp. 681-691, 1980.
25. Honig, M. L. and D. G. Messerschmitt, *Convergence Properties of an Adaptive Digital Lattice Filter*, IEEE Transactions on Circuits and System, Vol. CAS-28, pp. 482-493, Jun. 1981.
26. Gibson, C. and S. Haykin, *Nonstationary Learning Characteristics of Adaptive Lattice Filters*, Proceedings IEEE International Conference on Acoust., Speech and Signal Processing (Paris, May 1982), pp. 671-674.
27. Gibson, C. and S. Haykin, *A Comparison of Algorithms for the Calculation of Adaptive Lattice Filters*, Proceedings IEEE International Conference on Acoust., Speech and Signal Processing 80 (Denver, Colorado, Apr. 1980), pp. 978-983.
28. Makhoul, J., *New Lattice Methods for Linear Prediction*, 1976 IEEE International Conference on Acoust., Speech and Signal Processing, Philadelphia, pp. 462-469, Apr. 1976.
29. Hildebrand, F. B., *Introduction to Numerical Analysis*, New York, Toronto, London, McGraw-Hill Book Company, Inc., pp. 378-386, 1956.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2. Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3. Department Chairman, Code 62 Department of Electrical Engineering Naval Postgraduate School Monterey, CA 93943	1
4. Division of Personal Education and Training Republic of Korea Naval Headquarters Dae Bang Dong, Young Deung Po gu Seoul, Korea	2
5. Professor Murali Tummala, Code 62Tu Department of Electrical Engineering Naval Postgraduate School Monterey, CA 93943	6
6. Professor Sydney R. Parker, Code 62Px Department of Electrical Engineering Naval Postgraduate School Monterey, CA 93943	1
7. LCDR Ill K. Park, ROKN 650-369 Kwang Chun Dong, Kwang-Ju Seoul, Korea	10
8. Cheong, Hyun Min Department of Electronic Engineering Yonsei University, Seo Dae Mun Gu Seoul, Korea	1
9. Kim, Nam Soo Department of Electronic Engineering Yonsei University, Seo Dae Mun Gu Seoul, Korea	1

END

11-87

DTIC